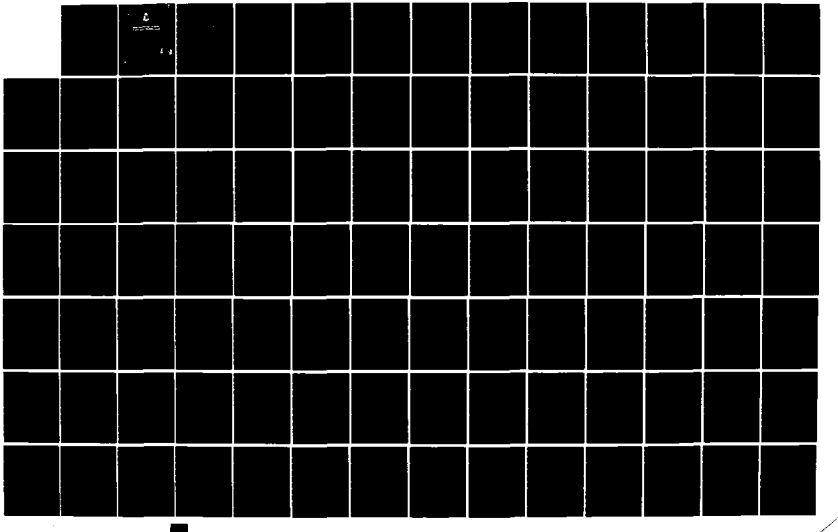
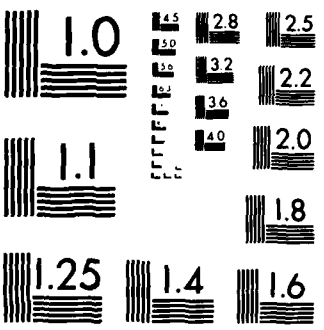
AD-A151 752 DESIGN SPECIFICATIONS FOR IMAGER: THE DREO (DEFENCE RESEARCH ESTABLISHMEN..(U) DEFENCE RESEARCH ESTABLISHMENT OTTAWA (ONTARIO) C BROCHU ET AL. SEP 83 1/2

UNCLASSIFIED DREO-TN-82-15 F/G 9/2 NL





National Defence / Défense nationale

DESIGN SPECIFICATIONS FOR IMAGER: THE DREO IMAGE PROCESSING SYSTEM

by

Claude Brochu

and

Roy Ball, Ken Lim, Soren Sorensen

DEFENCE RESEARCH ESTABLISHMENT OTTAWA

TECHNICAL NOTE 82-15

Canada

September 1983
Ottawa

National Defence / Défense nationale

DESIGN SPECIFICATIONS FOR IMAGER: THE DREO IMAGE PROCESSING SYSTEM

by

Claude Brochu
Electronics Division

and

Roy Ball, Ken Lim, Soren Sorensen
Roy Ball Associates Ltd

DEFENCE RESEARCH ESTABLISHMENT OTTAWA

TECHNICAL NOTE 82-15

PCN
33D

September 1983
Ottawa

ABSTRACT

This document enunciates the detailed design specifications for the development of image processing software (first three releases) for the DREO Image Processing System consisting of a Data General Eclipse S-130 computer equipped with a NORPAK RGP-3050 Image Processor. A Top-Down structured design technique approach is used. It is composed of data and file formats, structure charts and programming specifications for all modules. These modules allow image transfer and manipulation, system parameters bookkeeping and multispectral and monospectral image analysis techniques.

RÉSUMÉ

Ce document énonce les caractéristiques détaillées pour la conception d'un logiciel de traitement numérique d'images (3 premières éditions) pour le Système de traitement d'images du CRDO comprenant un ordinateur de Data General l'Eclipse S-130 et un système d'affichage et de traitement video de NORPAK. Une technique de conception structurée est employée. Cette description comprend les formats des données et des fichiers sur bande et sur disque, les diagrammes structurés de conception et les spécifications pour programmer chacun des modules. Ces divers modules permettent le transfert d'images entre les différents supports périphériques, la mise à jour et la connaisance des paramètres du système et l'analyse numérique par différentes techniques d'images multispectrales et monospectrales.

TABLE OF CONTENTS

TABLE OF CONTENTS (CONT'D)

TABLE OF CONTENTS (CONT'D)

TABLE OF CONTENTS (CONT'D)

TABLE OF CONTENTS (CONT'D)

LIST OF TABLES

RELEASE NOTE

The current implementation of IMAGER introduces some new features:

(1) A Monospectral Analysis Package has been added at the main menu level. This package allows the operator to perform any of the following operations on an image: histogram equalization, magnification (zoom), colouration via level slice, and a dump of the pixel intensities of a specified region.

(2) Two new features are accessible at the Multi-Spectral Analysis menu level:

   (a) The algebriac combinations feature enables the operator to compute a new image using an algebraic combination of existing images.

   (b) The colour composite feature is useful when comparing up to three registered images taken through different multi-spectral bands.

(3) The refresh-to-disk feature, accessible at the Route Images menu level, allows the operator to copy the contents of a NORPAK refresh buffer to a disk image file.

(4) IMAGER now creates and builds a log file, IMAGER.LO. Entries are added to this log file for each image file created during an IMAGER session by the following operations: ERIM tape to disk, refresh to disk, algebraic combinations, Gaussian classifier or ratio. The operator can look at the log file using a new option of the system parameters list and update menu. In addition, when this file is no longer of interest, the operator can delete it and start a new log file.

(5) The header of a standard IMAGER file now has a different format. The new layout is given in Section 3.2.1.

1.0 SCOPE

This design specification defines the software for the 3 baseline releases of the IMAGER project being developed at the Defence Research Establishment Ottawa (DREO).

2.0 APPLICABLE DOCUMENTS

1. Software Control Plan Rev. 2.0 internal paper.
2. Requirement Specification for IMAGER. The DREO Image processing system, DREO Tech Note 82-14.
3. NORPAK Programmer's reference mannual.

3.0 OVERALL DESIGN

As baseline 3 is essentially an enhanced version of baseline 2, most of the changes are implemented by adding several new modules. Some existing baseline 1 and 2 modules are modified as well to reflect the new implementation.

Baseline 1 and 2 modules that have changed are: IMAGER, INITIAL (formerly INIT), MENU, RTIMG, RDERM, RDKRF, EDIT, IMGOP, RRFMO (formerly RRFDP), SPLU, LIST, MSA, RATIO, CLASS.

New baseline 3 modules are SEQNO, RRFDK, LOGLS, COPH, COLCO, ALGCOMB, EQINPUT, RPOLISH, COMPILE, EQPROC, SSA, RBOLD, HISTO, TRANS, EQUIL, HPLOT, GRAPH, SLICE, SSLICE, LSLICE, ZOOM, ZOOMI, MAGNI, PIXCO, DDUMP, STATS, PDATA, PHEAD and MSG.

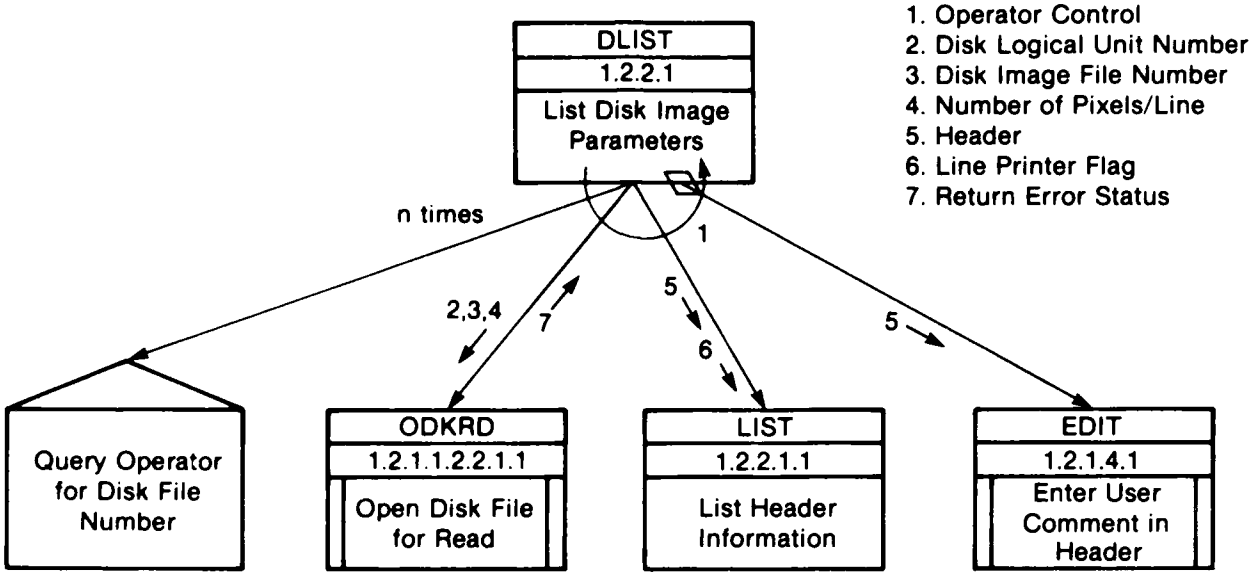
The overall design is contained in three subsections. Section 3.1 details the layout of the Imager task using structure charts. Section 3.2 describes the data structures used by IMAGER. Section 3.3 contains the module specifications for all IMAGER modules, new or existing. Some of the HIPO's written for the baseline 1 software may not conform to HIPO specifications as described in the software control plan. This is because the Software Control Plan was produced after the IMAGER software was started.

3.1 Program Hierarchy

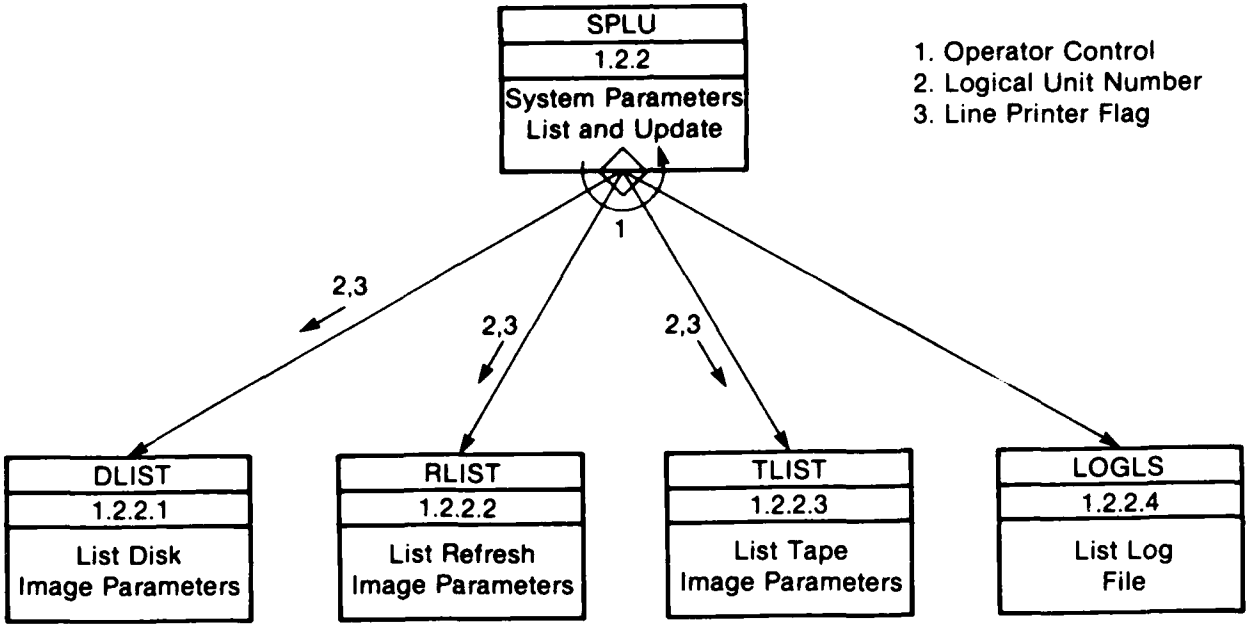
The following structure charts illustrate the organization of the IMAGER task. The modules indicated are all part of the IMAGER system. FORTRAN and NORPAK utilities are intentionally omitted from the diagrams.

A list of the charts follows.

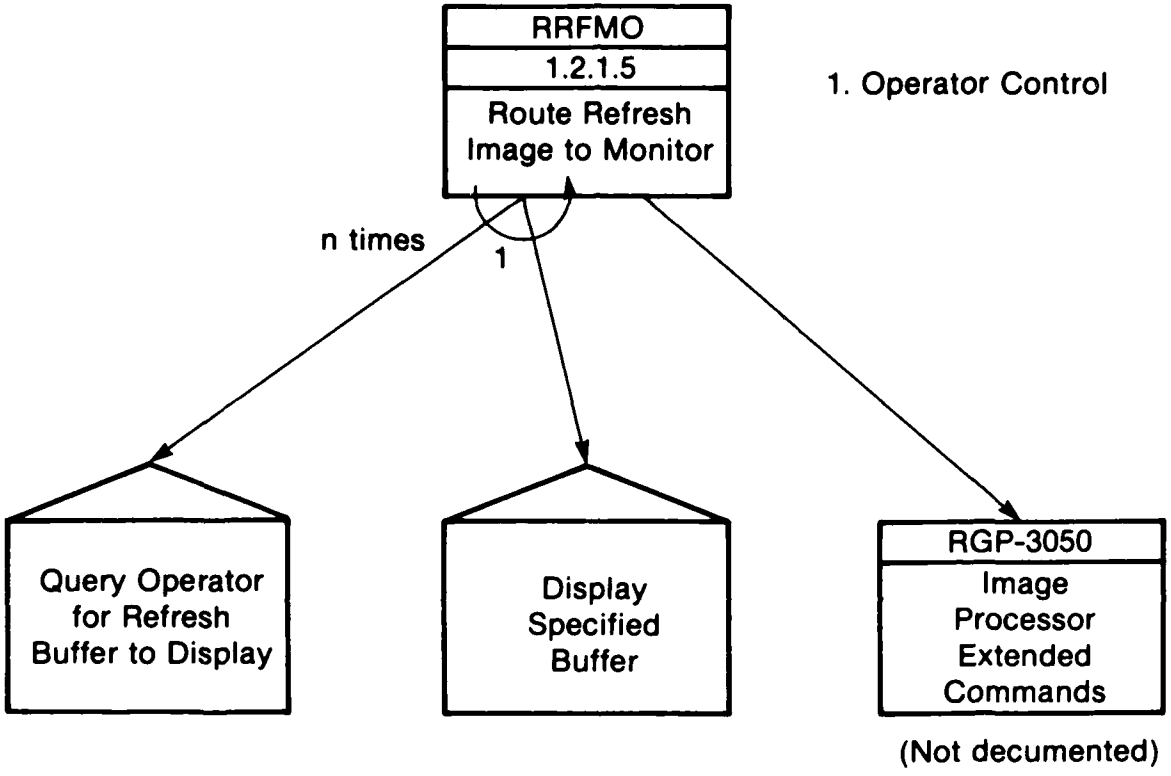
3.1.1 Structure of the IMAGER task



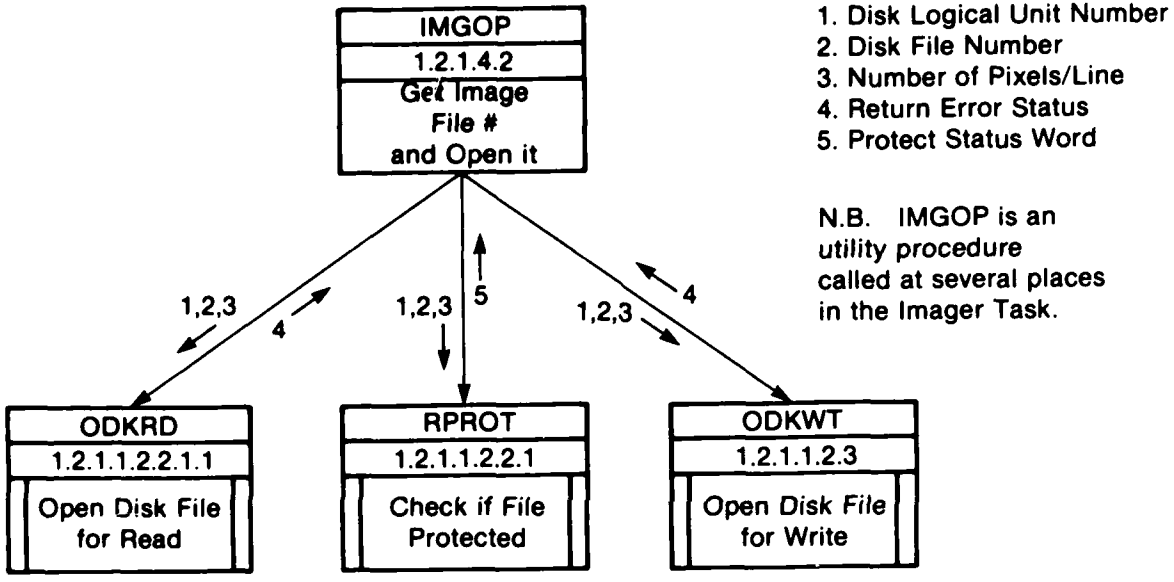
3.1.13 Structure of the DLIST procedure



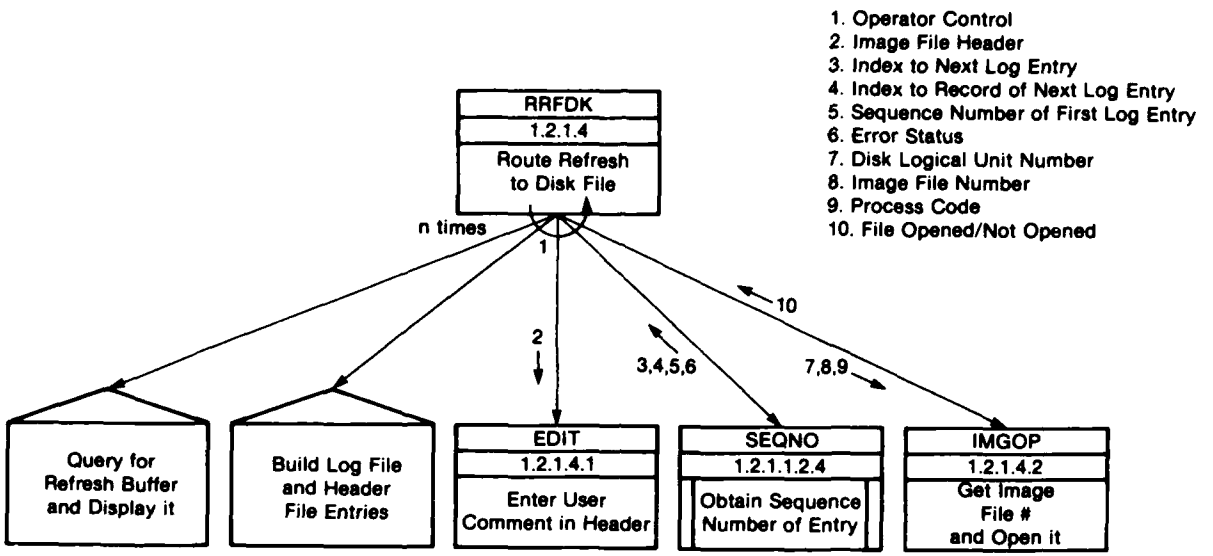
3.1.12 Structure of the SPLU procedure



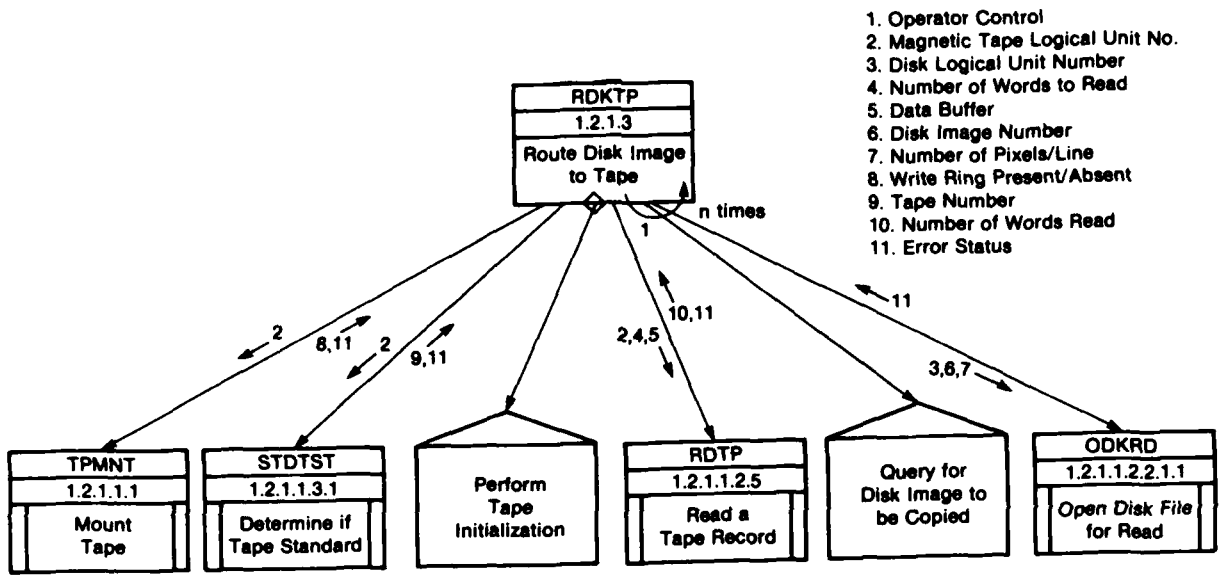
3.1.11 Structure of the RRFMO procedure



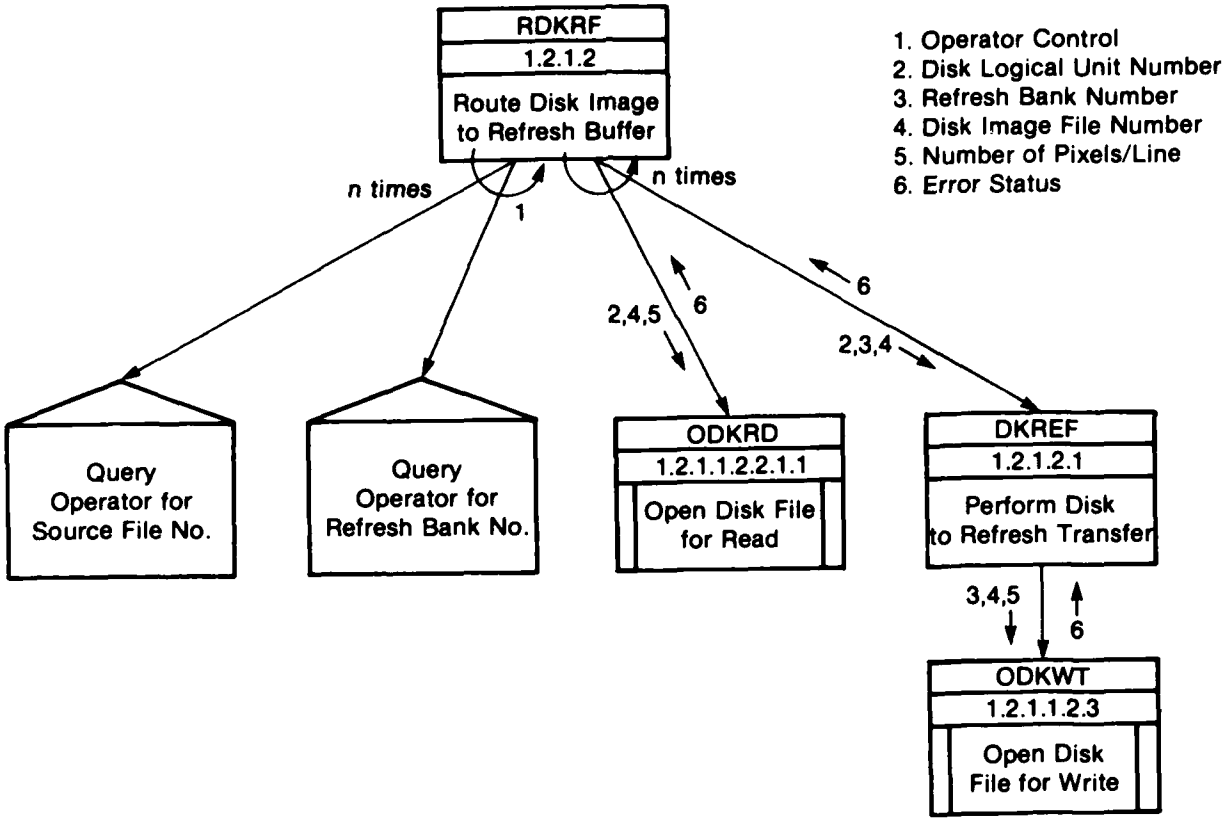
3.1.10 Structure of the IMGOP procedure



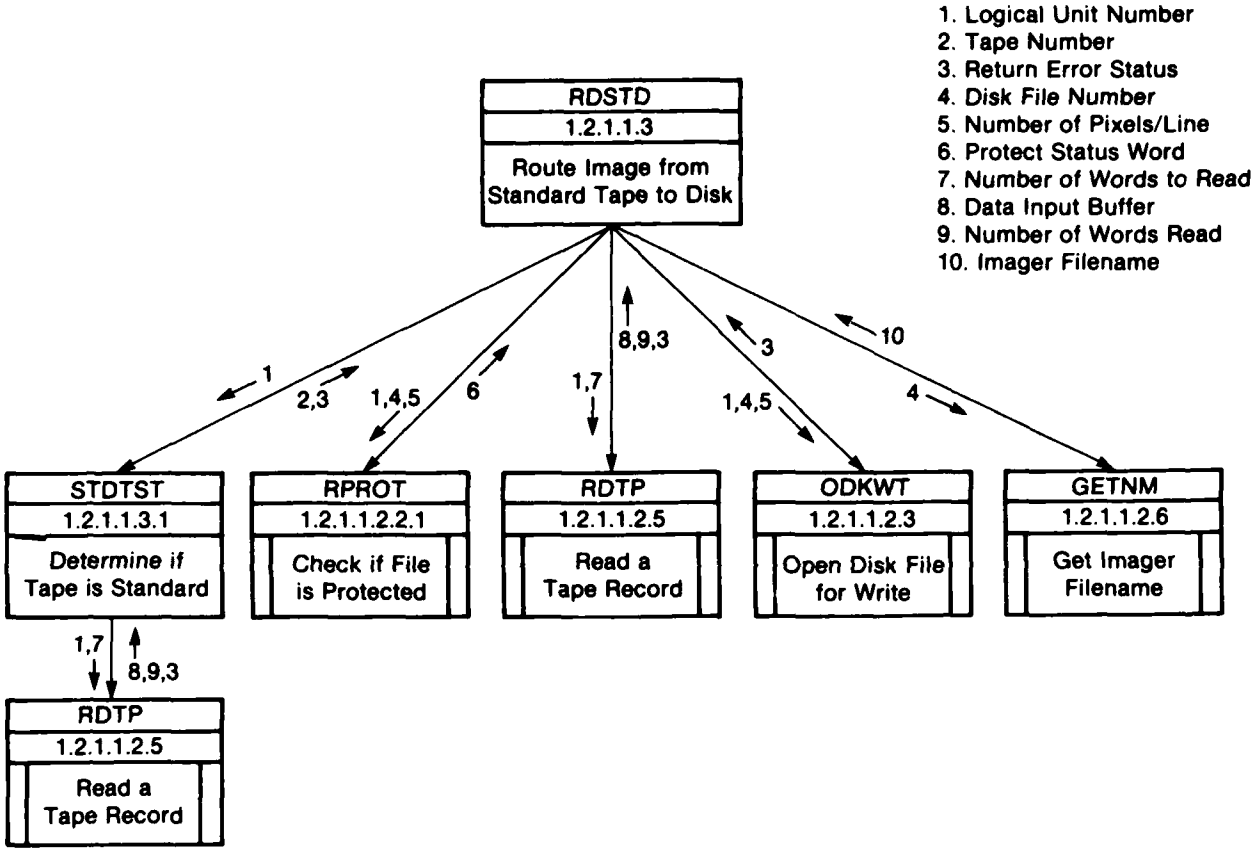
3.1.9 Structure of the RRFDK procedure



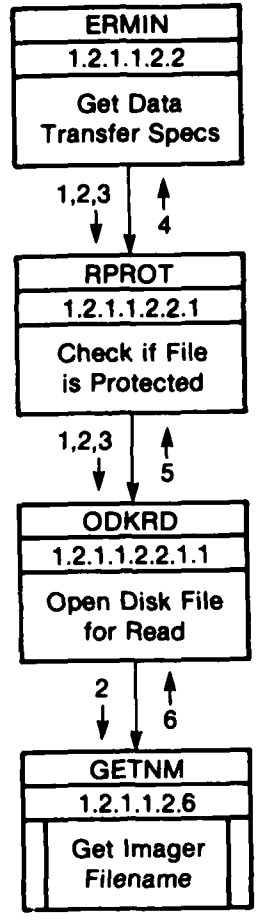
3.1.8 Structure of the RDKTP procedure



3.1.7 Structure of the RDKRF procedure



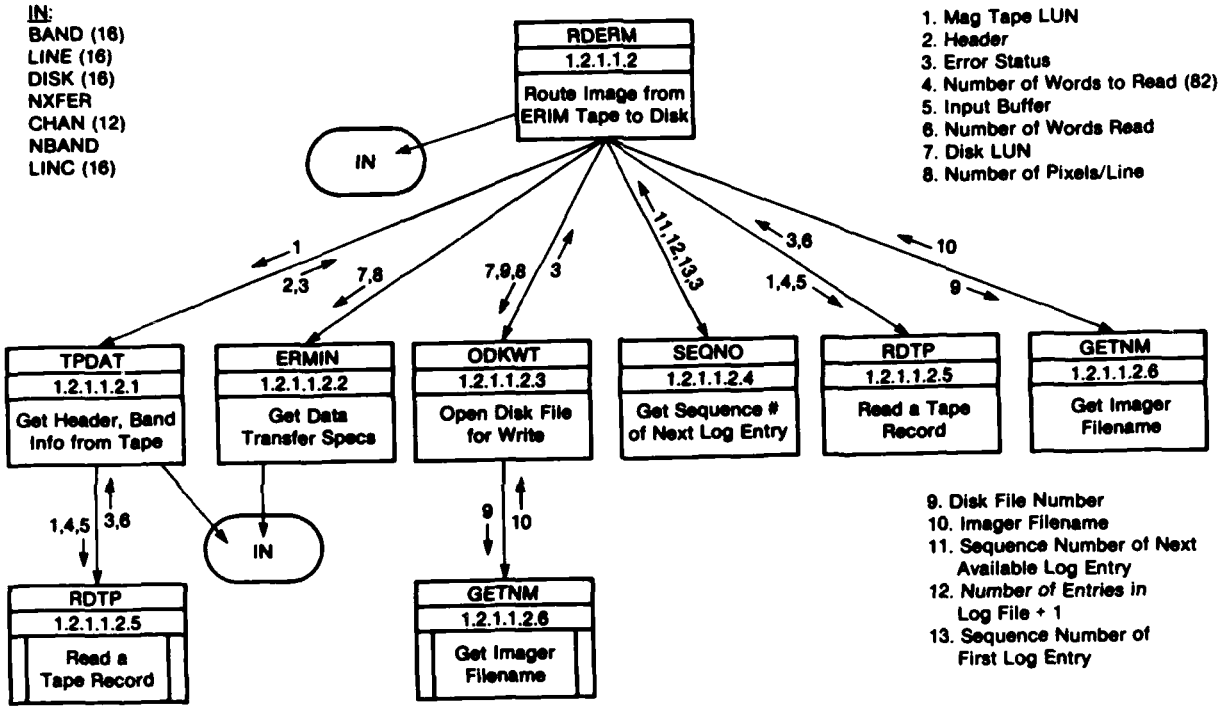
3.1.6 Structure of the RDSTD procedure



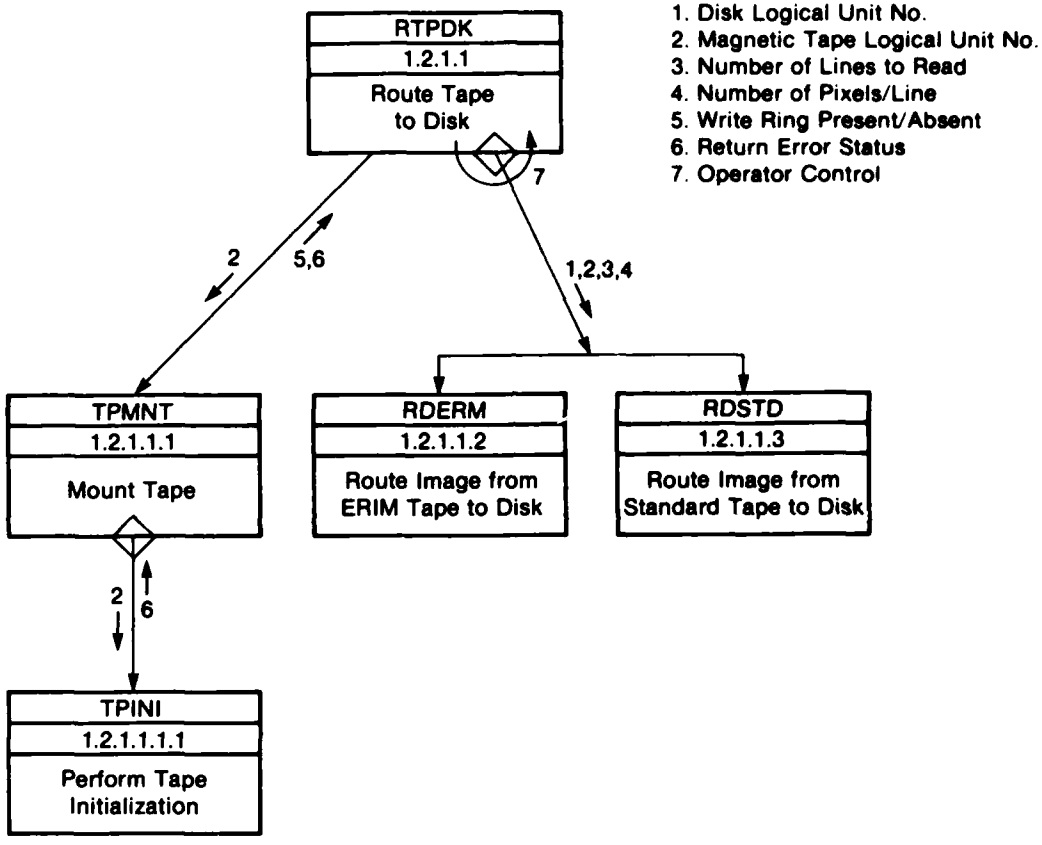
1. Disk Logical Unit Number
2. Disk File Number
3. Number of Pixels/Line
4. Protect Status Word
5. Error Status
6. Imager Filename

N.B. RPROT is an utility procedure called at several places in the Imager Task.

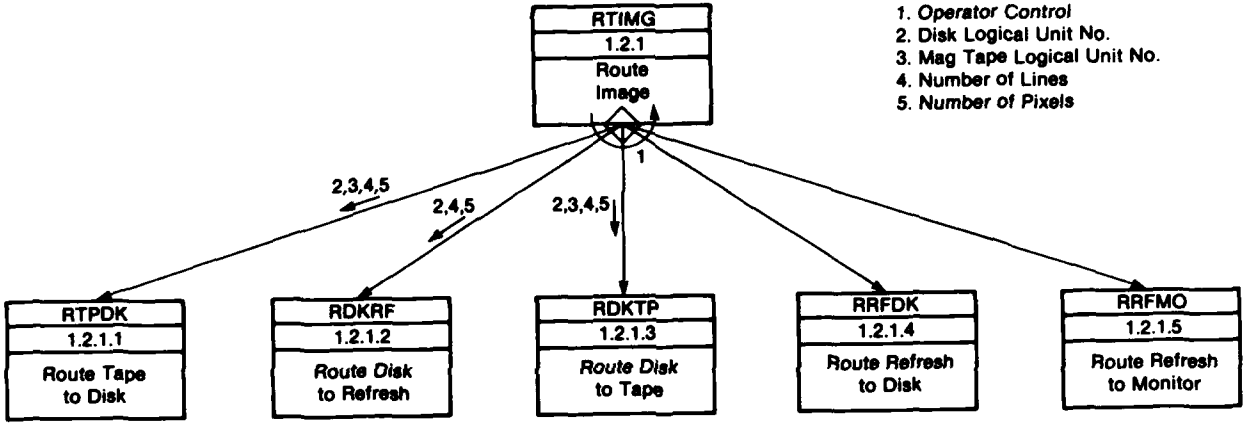
3.1.5 Structure of the RPROT procedure



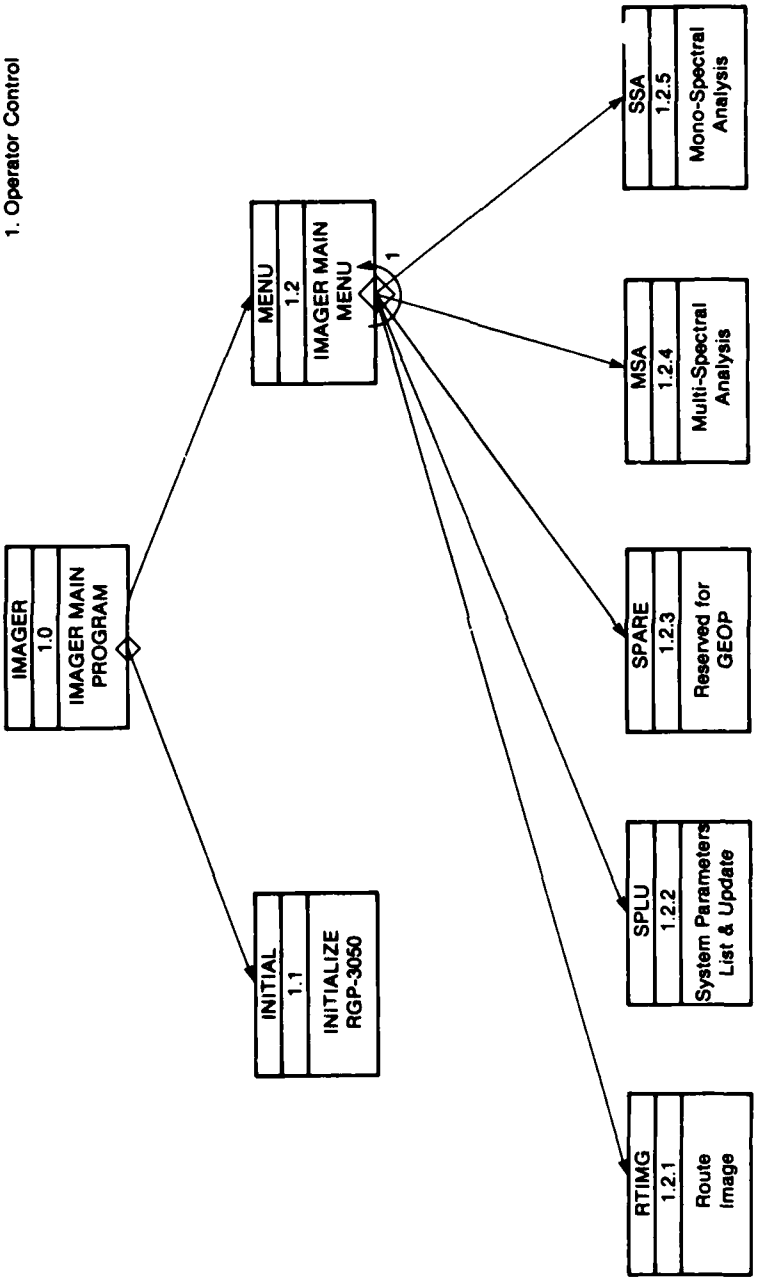
3.1.4 Structure of the RDERM procedure



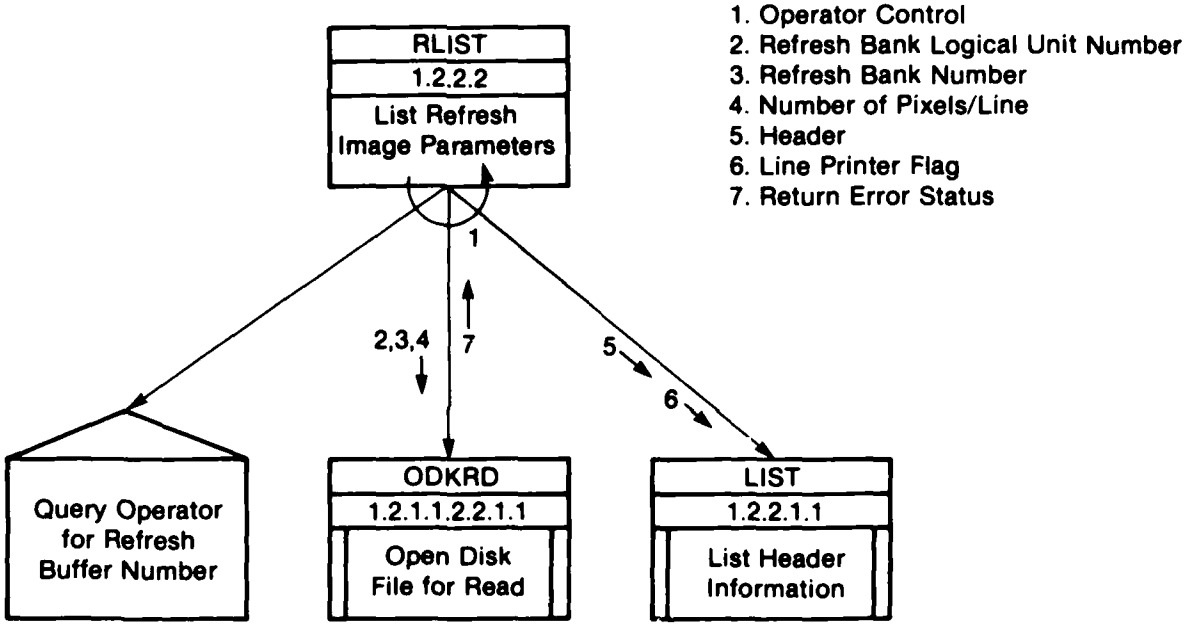
3.1.3 Structure of the RTPDK procedure



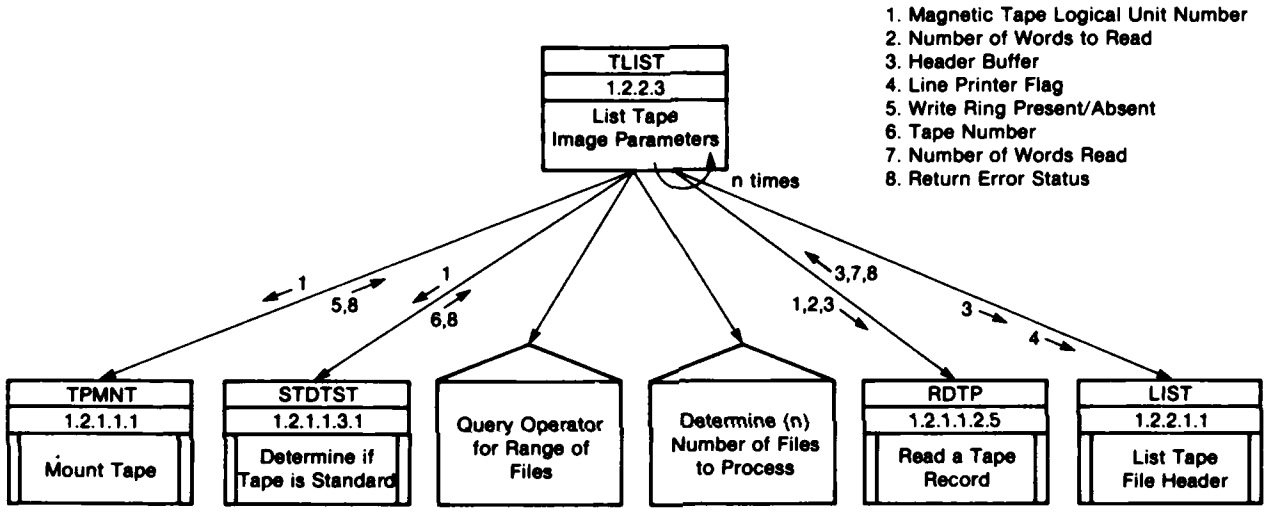
3.1.2 Structure of the RTIMG procedure



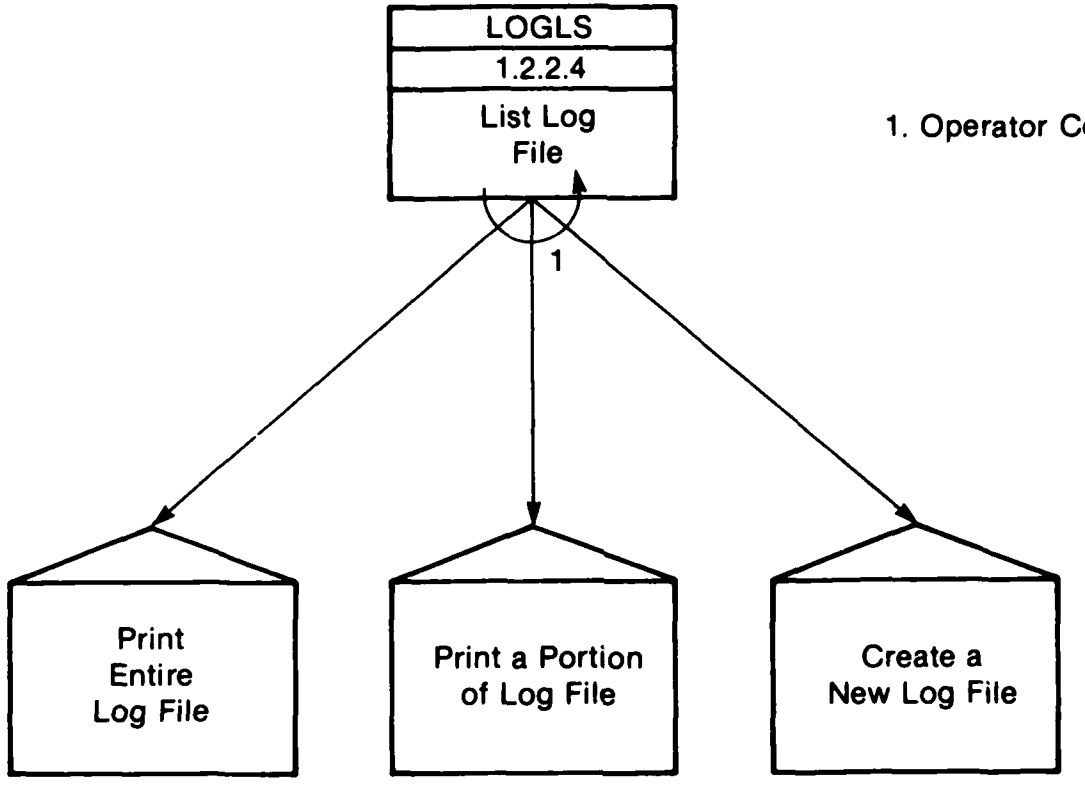
3.1.1 Structure of the IMAGER task



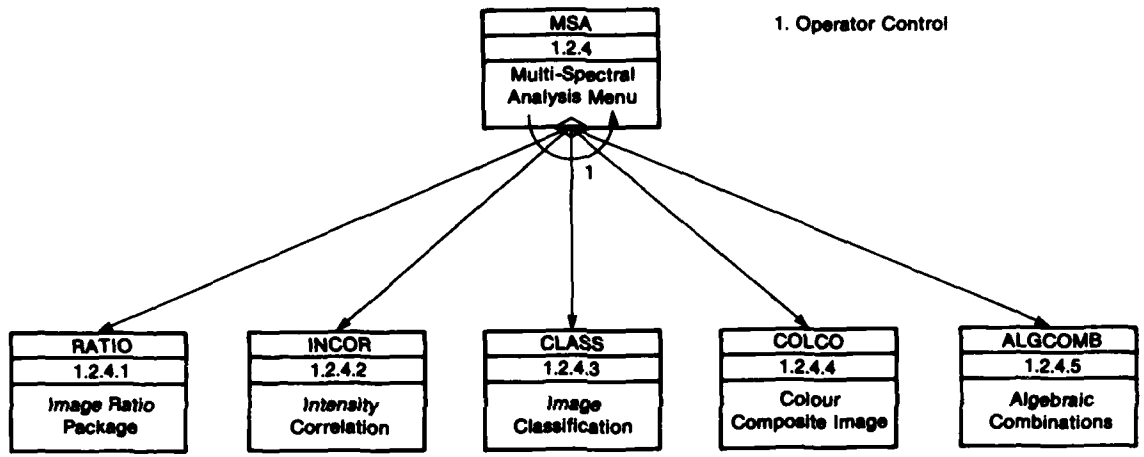
3.1.14 Structure of the RLIST procedure



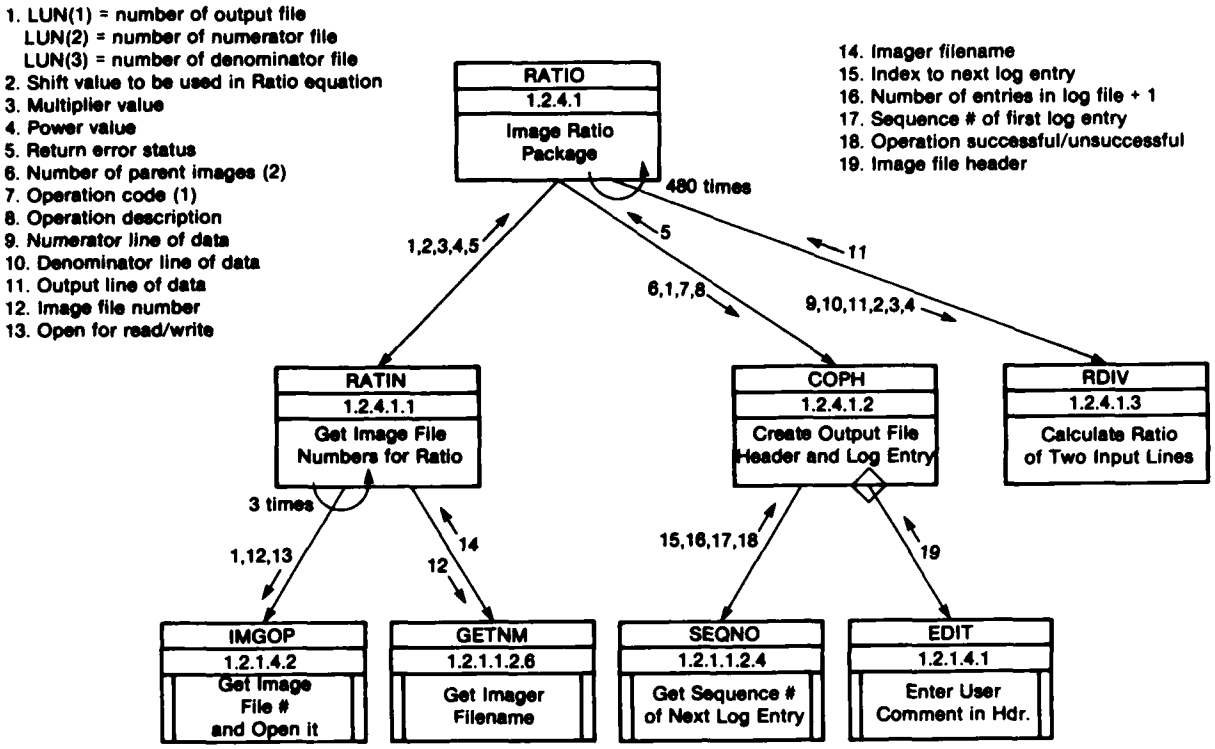
3.1.15 Structure of the TLIST procedure



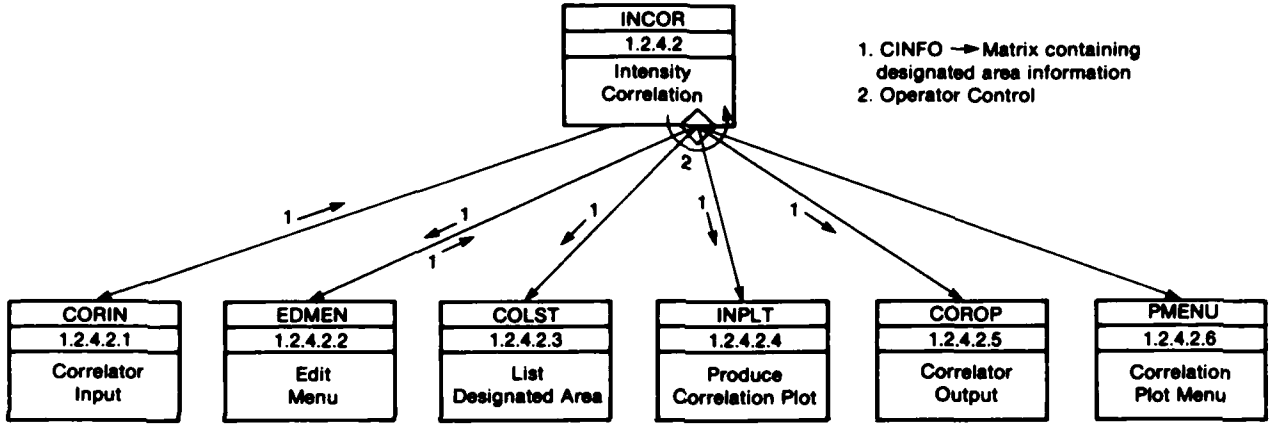
3.1.16 Structure of the LOGLS procedure



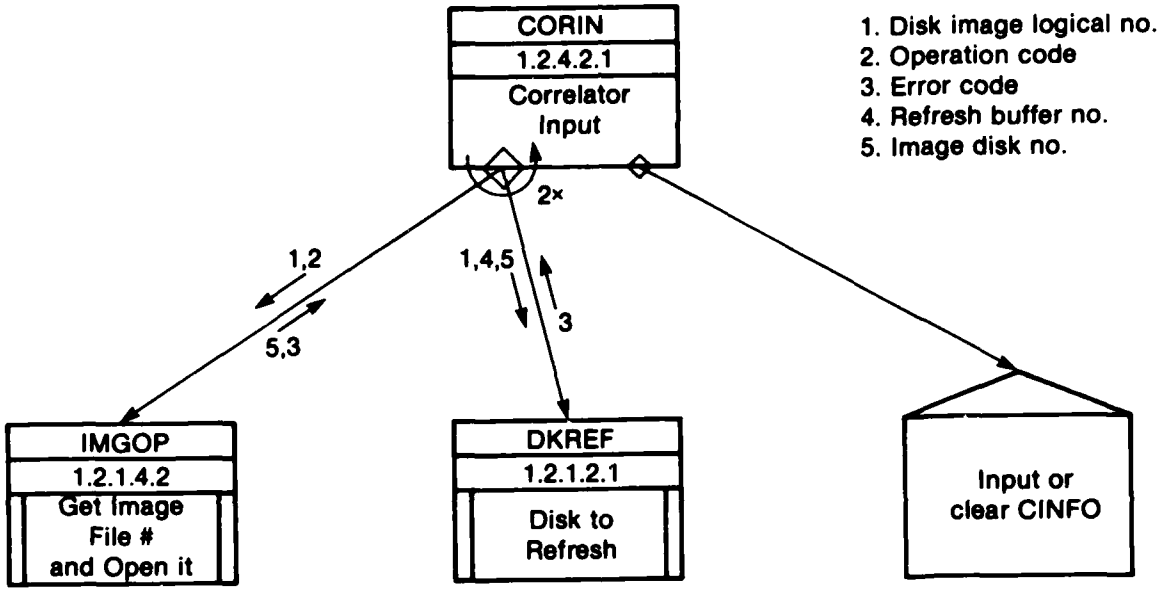
3.1.17 Structure of the MSA procedure



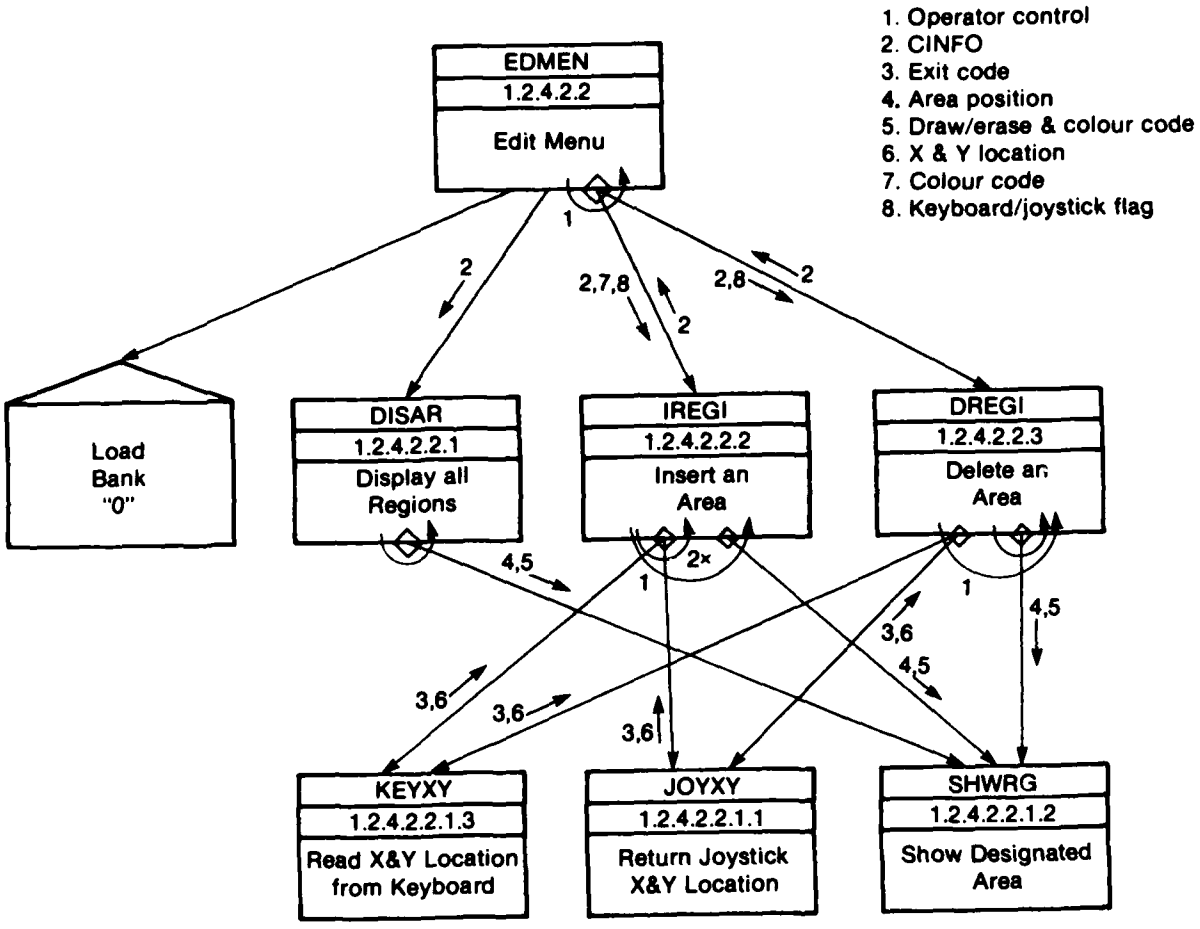
3.1.18 Structure of the RATIO procedure



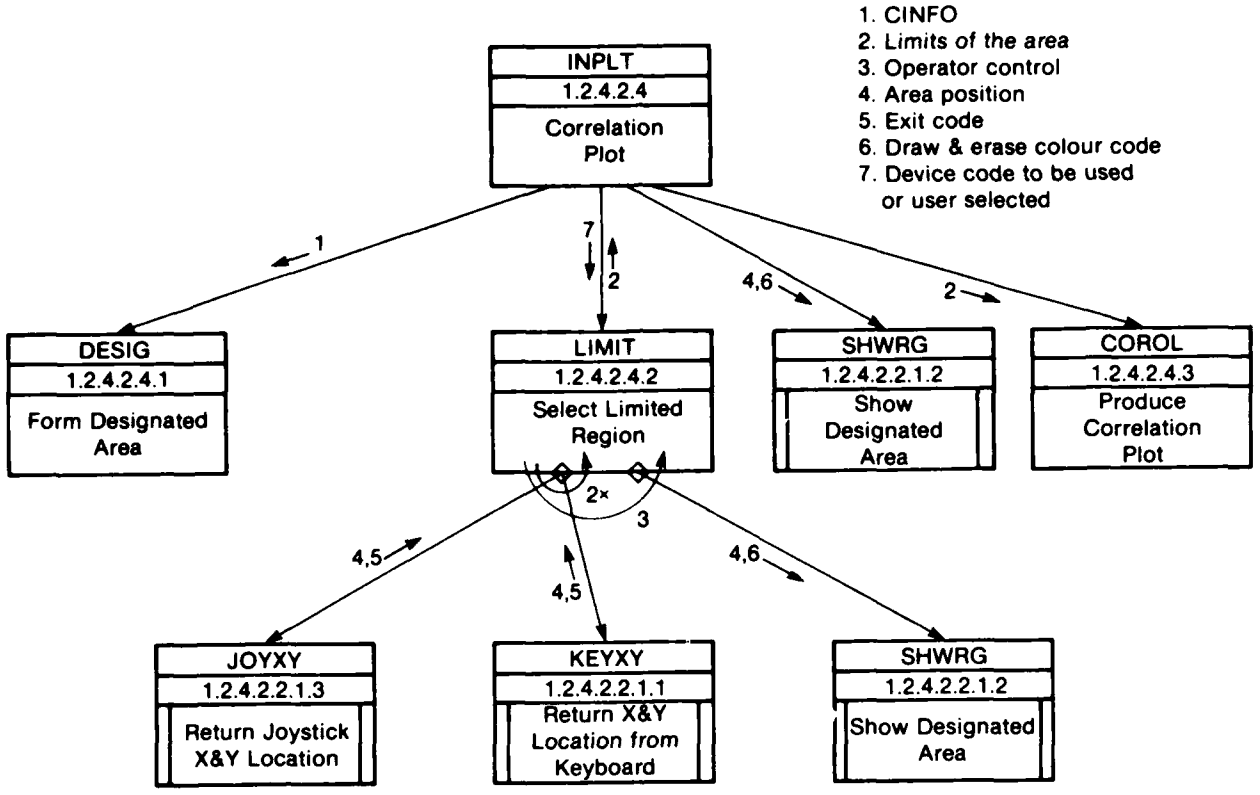
3.1.19 Structure of the INCOR procedure



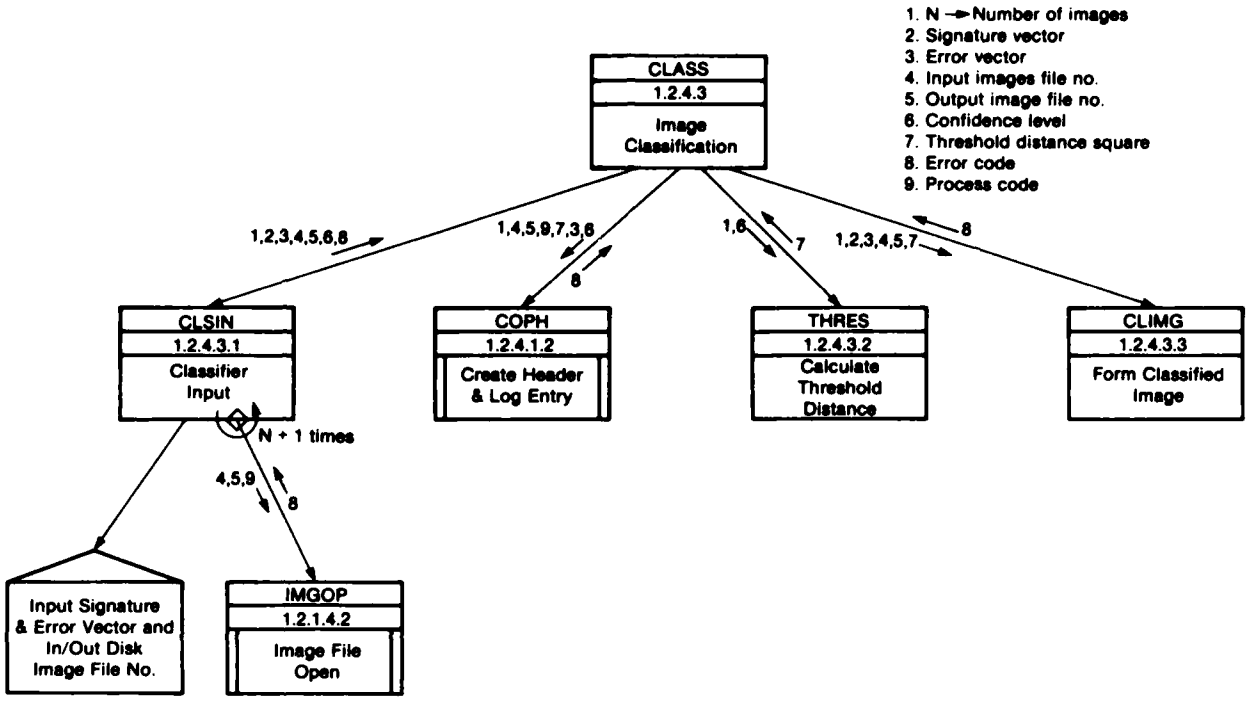
3.1.20 Strucutre of the CORIN procedure



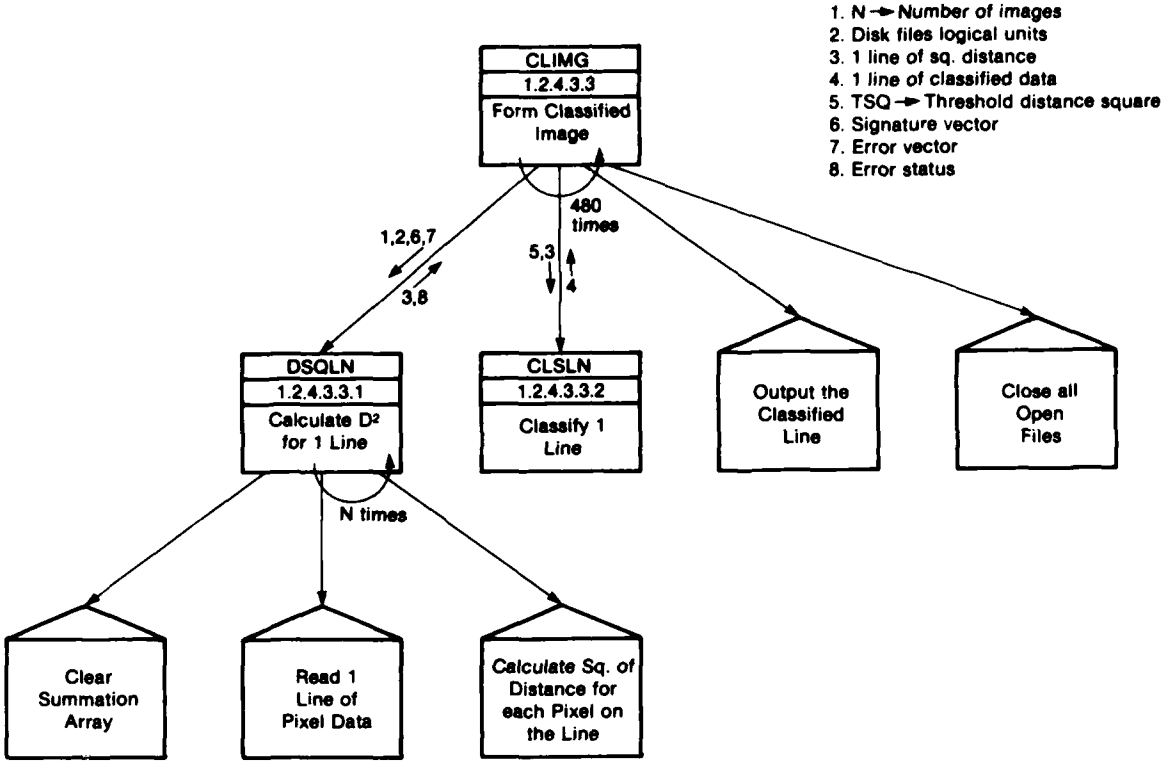
3.1.21 Structure of the EDMEN procedure



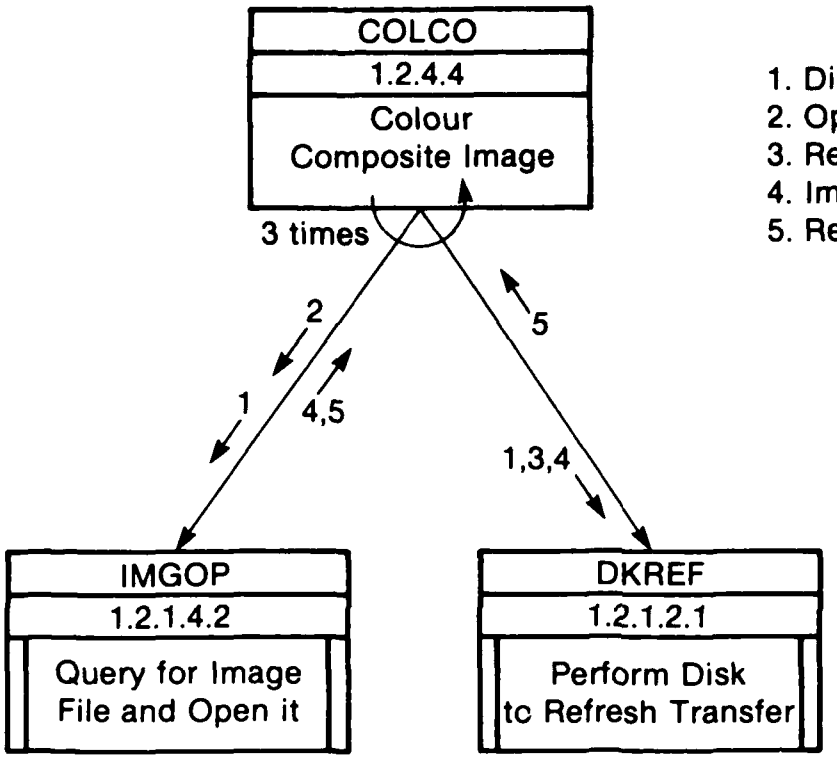
3.1.22 Structure of the INPLT procedure



3.1.23 Structure of the CLASS procedure

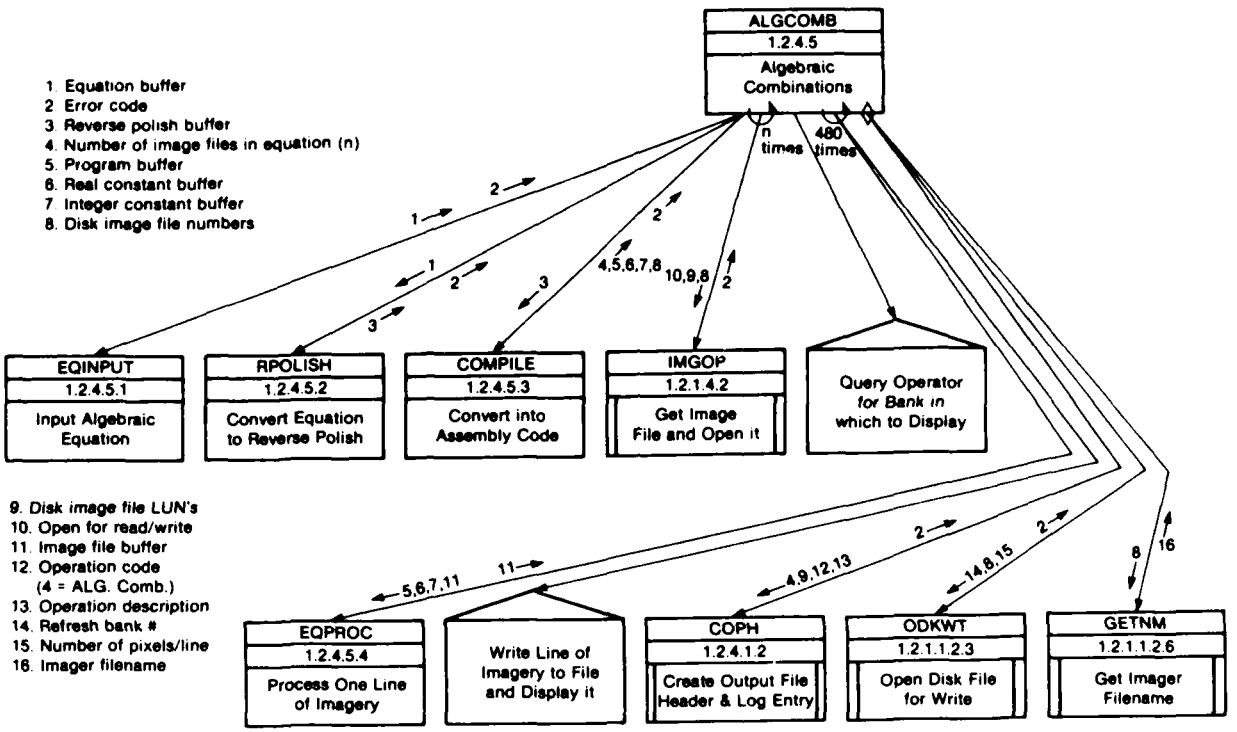


3.1.24 Structure of the CLIMG procedure

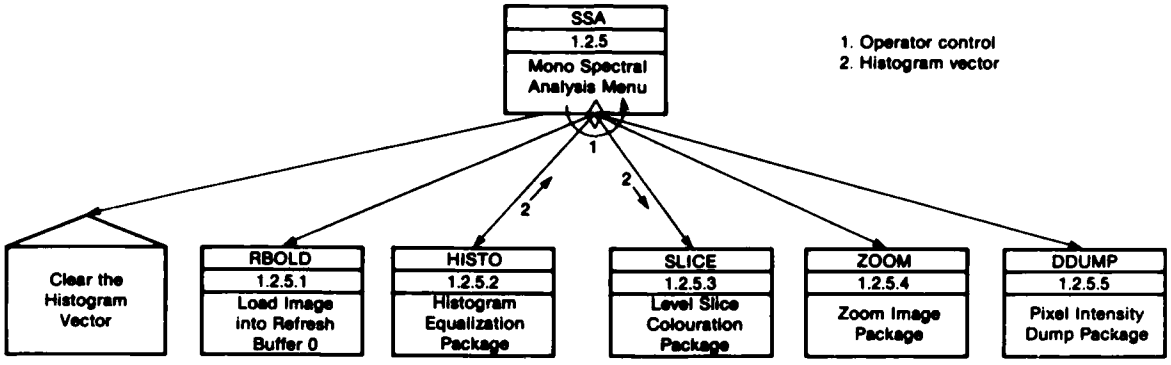


1. Disk logical unit number
2. Open for read/open for write
3. Refresh bank number
4. Image file number
5. Return error status

3.1.25 Structure of the COLCO procedure



3.1.26 Structure of the ALGCOMB procedure



3.1.27 Structure of the SSA procedure

TABLE 5

LOG FILE RECORD FORMAT

WORD	VARIABLE DESCRIPTION	VARIABLE TYPE
1	Sequence Number of this entry	Integer
2	Month	Integer
3	Day	Integer
4	Year	Integer
5	Hour	Integer
6	Minute	Integer
7	Second	Integer
107-156	User Comments	ASCII
157	Number of Parent Images (1-6)	Integer
158	1st Parent Sequence No. or External Tape No. (2 words)	Integer
159	2nd Parent Sequence No.	Integer
160	3rd Parent Sequence No.	Integer
161	4th Parent Sequence No.	Integer
162	5th Parent Sequence No.	Integer
163	6th Parent Sequence No.	Integer
164	Operation Code	Integer
165-256	Operation description	---

Word 164 = 0 → ERIM tape to disk
= 1 → Created by RATIO
= 2 → Created by CLASS
= 3 → Created by RRFDK
= 4 → Created by ALGCOMB

When word 164 is 0, words 158 to 159 contain 4 ASCII characters representing the external tape number.

The field layout for a log entry is quite similar to the information contained in the header record for an image file. The log entry, however, contains more information concerning the creation of the new image. See Tables 5-10 below for the layout.

The existing modules RDERM, CLASS and RATIO, since they create disk images, must be modified to create the log entries as well. The new modules RRFDK and ALGCOMB will perform this function as well.

Utilities that the above routines will involve are:

(a) SEQNO - this new utility returns the values NXTSEQ, NXTREC and FRSTSEQ to the caller and then increments NXTSEQ and NXTREC.

(b) COPH - this utility must be modified to create the log entry as well as the header record.

The new module LOGLS will format and list the entire log or a portion of it on the line printer. Alternatively it will delete IMAGER.LO and then begin a new version of the file.

3.2.4 The IMAGER Message File

Because of the large quantity of messages the IMAGER program needs for its operation, namely the menus, queries and error messages, too much computer memory was required to store them. In order to minimize the memory requirement of the IMAGER software so that expansion becomes easier and more memory is made available for program and data, a message file system is designed to contain most of the text and output FORMATs specifications required.

This message file IMAGER.ER, organized as a random access file, has a record length of 256 Bytes. Table 11 shows the format of its record.

The format specifications are first entered into a sequential file MSGE via a text editor and are later processed by a special utility program BERRO to produce IMAGER.ER. BERRO prints also on line printer a list of all messages along with their sequence numbers. The subroutine MSG (described in 3.3.119) used to output these messages requires these numbers for its operation.

3.2.3 The IMAGER Log File

IMAGER keeps a log file of all disk images created during an IMAGER session. Currently there are five operations that create a new disk image: refresh to disk, ERIM tape to disk, Gaussian classification of images, ratio of two images, and algebraic combination of images. When one of these operations creates a disk image, it must also perform the following actions:

- allocate an internal 256 word log entry buffer,
- set up specific fields of this buffer with information concerning the origin of this new image (i.e. data, time, comments, operation code, etc.),
- assign the next available log sequence number to this new entry and display it to the operator,
- append this log entry to the end of the existing log file,
- increment the next available log sequence number.

The log file has the filename IMAGER.LO. It consists of a header record followed by log entry records created as described above.

The header record contains the following information in the first three words:

WORD 1 - NXTSEQ - The next available log sequence number

WORD 2 - NXTREC - The total number of records in the log file + 1 (including the header record). This value is also the index to the next available record number.

WORD 3 - FRSTSEQ- The sequence number of the first log entry.

Note: Words 4-256 inclusive of the header are unused.

The use of the log sequence number and an explanation of the header record is best given using an example. Suppose a header record contains the following values:

NXTSEQ = 120
NXTREC = 7
FRSTSEQ = 115

This log file consists of the above header record followed by the log entries indexed by the consecutive numbers 115, 116, 117, 118, and 119 respectively. The sequence number 120 will be assigned to the next log entry when it is created and when this is done, the newly created entry will be the 7th record in the file (including the header).

When the log file is deleted and a new one created (see LOGLS, below), the value for NXTSEQ is retained and copied into FRSTSEQ rather than beginning again at sequence number 1.

Standard IMAGER Format

This format applies to all disk files and digital magnetic tapes written by the Imager system. Each standard image file (either on tape or disk) contains one band of an image 480 lines by 640 pixels. The size of each file is 481 records of 640 pixels. The first record is the header record whose layout is shown in Table 4. The subsequent 480 records each contain one line of the image stored from left to right and from the top to the bottom.

A standard format tape contains a number of these files with two end of file (EOF) marks terminating the tape. In order for the Imager system to output data to a tape, that tape must first be fully initialized through the Command Line Interpreter (CLI) of the RDOS system. IMAGER has no capability of overwriting existing tape files.

3.2.2 MSA Data Structure for Image Correlation (CINFO)

INCOR will require one data array. This array will be named CINFO and contained the designated area information. CINFO will be a three-dimentional matrix 3 by 4 by 10 elements. The first dimension of 4 elements will index the spacial position of the designated areas; in order Xmin, Xmax, Ymin, Ymax, zeros assigned to Xmin signify an unused position. The second dimension of 3 elements will index the colour of the designaed area; in order red (1), green (2), blue (3). The third dimension of 10 elements will index the storage for up to 10 designated areas per colour.

On operator request CINFO will be read and stored on a disk file, this file will contain 120 records of one word each and will be accessed from/to the disk as an entity. This file name will have a ".CO" extension.

TABLE 4

IMAGER STANDARD HEADER FORMAT

WORD	VARIABLE DESCRIPTION	VARIABLE TYPE
1	Write protect flag or standard tape file number	Integer
2	Sequence Number	Integer
3	Month	Integer
4	Day	Integer
5	Year	Integer
6	Hour	Integer
7	Minute	Integer
8	Second	Integer
9-107	Spare	---
108-157	50W (100 characters) of User Comments	ASCII
158	Spare	---
159-160	Source tape identification for ERIM tapes	ASCII
161-320	Spare	---

TABLE 2

ERIM HEADER (REFORMATTED)

BYTE	DESCRIPTION	TYPE
1-4	Scanner name	ASCII
5-6	Scanner unit number, numerals	ASCII
7-14	Data of flight (bDDMMMYY)	ASCII
15-16	Blank	ASCII
17-20	4 character Xerox tape no.	ASCII
21-38	Blank	ASCII
39-42	Heading in degrees, right justified	ASCII
43-48	Alitude in feet, right justified	ASCII
49-54	V/H ratio	ASCII
55-58	Run number, righ justified	ASCII
59-62	Mission number, right justified	ASCII
63-66	Tape number, right justified	ASCII
67-138	Description of scene	ASCII
139-140	Number of bands	INTEGER
141-164	Band numbers	INTEGER

TABLE 3

ERIM RECORD FORMAT

BYTE	DESCRIPTION	FORMAT
1-2	Band number	Integer
3-4	Line number	Integer
5-651	647 pixels	Integer
625-795	Scanner Information	Integer
796-808	Trailer Information	Integer

3.2 Data Structures

3.2.1 Formats for Tape and Disk Files.

There are three different formats which are of concern. These are:

1. Original ERIM format.
2. IMAGER ERIM format (reformatted).
3. IMAGER Standard format.

The original data is stored on digital magnetic tapes in the original ERIM format. Although these data tapes are not used as input to the IMAGER system, their format is described here briefly. This format is a pixel interleave with each tape record being 9900 bytes in length and containing all twelve bands of one image line. Each tape contains only one very large file containing the complete image. The first record on each tape is the header which is 138 bytes in length. The information contained in the header is shown in Table 1.

IMAGER ERIM Format

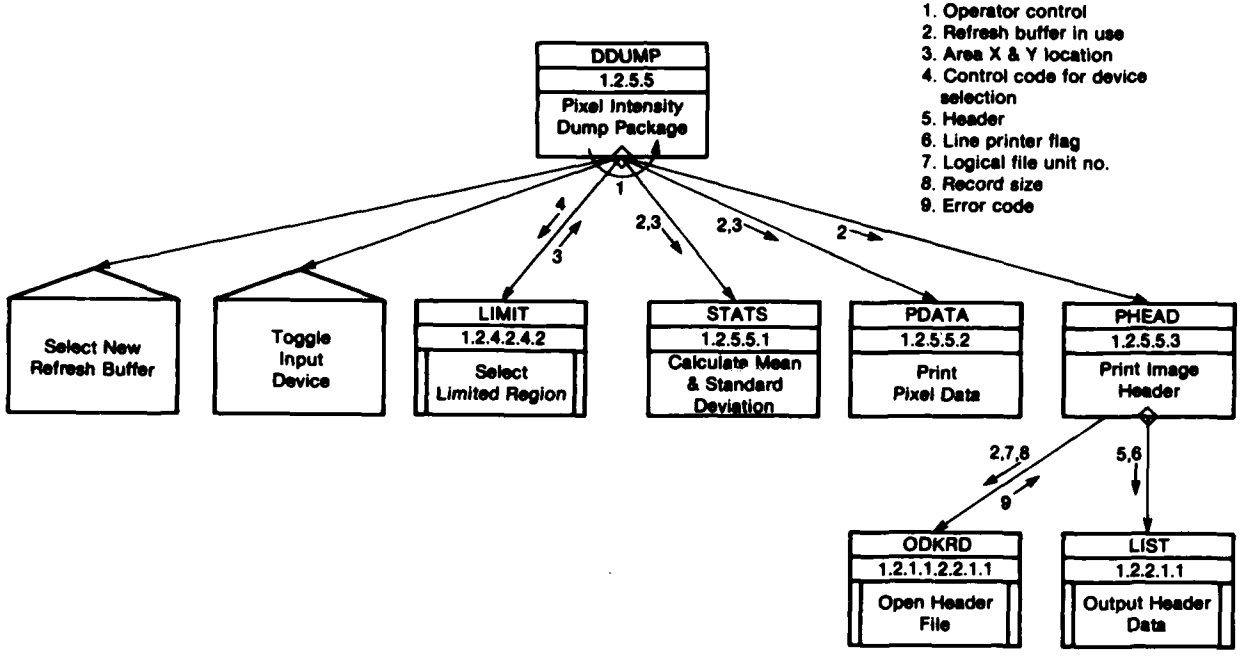
Due to the fact that the Data General Eclipse system has the capability of reading tape records of up to a maximum of 8192 bytes, the original ERIM format tapes had to be reformatted. The reformatting was performed on the Xerox Sigma system and the details will not be discussed in this documentation. The IMAGER ERIM format tapes (to be called ERIM format) each consist of one very large file containing the image.

The first record is 164 bytes in length while all others are 808 bytes. The first record is the header record containing the information shown in Table 2. The other records each contain one band of an image line and are structured as shown in Table 3. The tape records are ordered such that all bands of one image line precede the bands of the next image line.

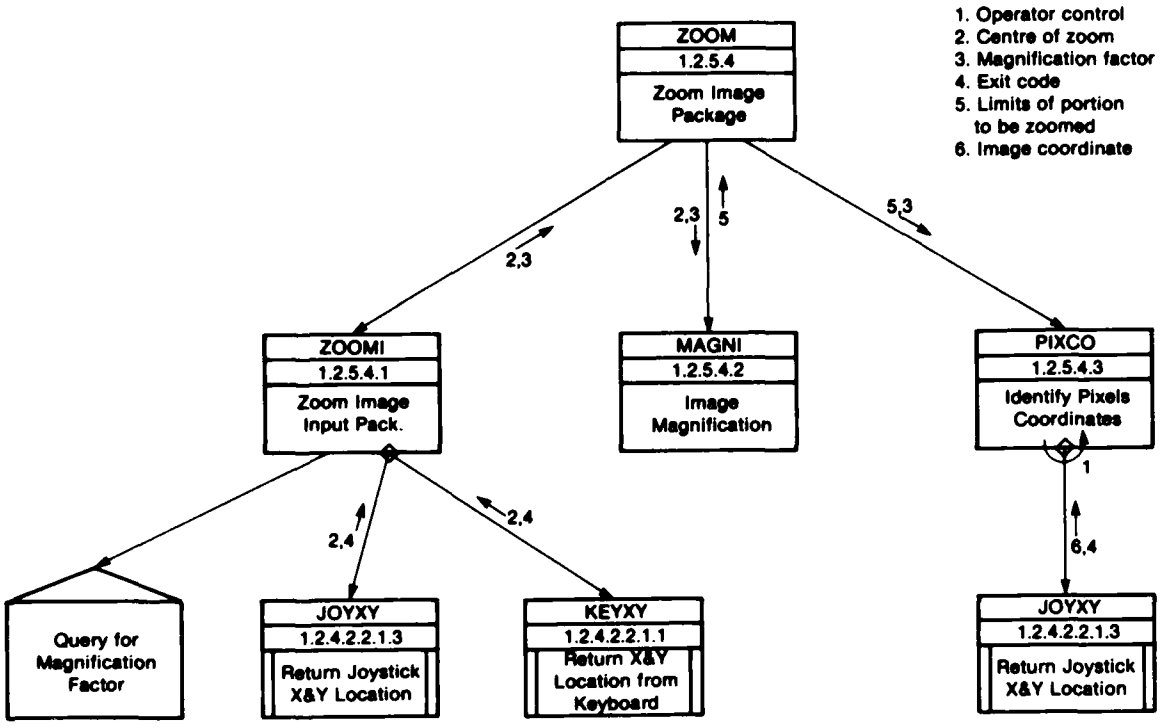
TABLE 1

ORIGINAL ERIM HEADER

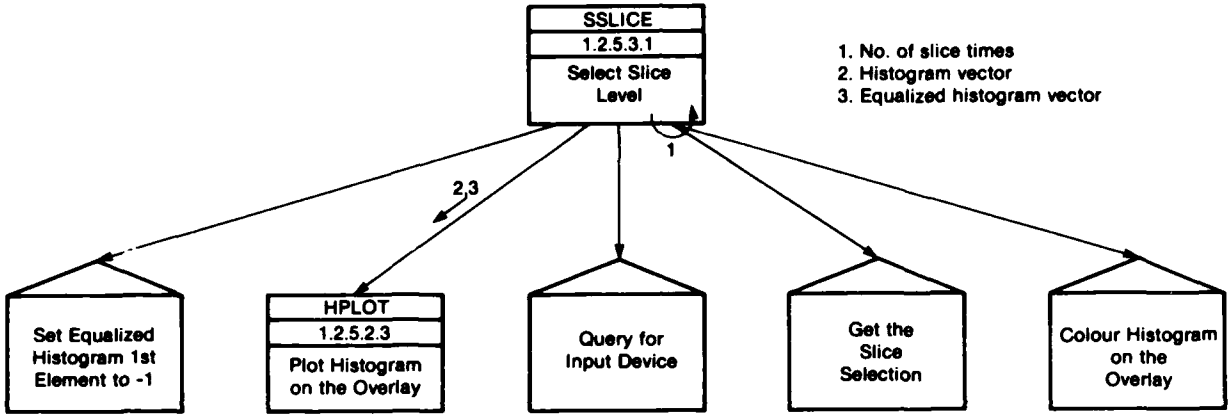
BYTE	DESCRIPTION	TYPE
1-4	Scanner name	ASCII
5-6	Scanner unit number, numerals	ASCII
7-14	Date of flight (DDMMMYY)	ASCII
15-38	Blank	ASCII
39-42	Heading in degrees, right justified	ASCII
43-48	Altitude in feet, right justified	ASCII
49-54	V/H ratio	ASCII
55-58	Run number, right justified	ASCII
59-62	Mission number, right justified	ASCII
63-66	Tape number, right justified	ASCII
67-138	Description of scene	ASCII



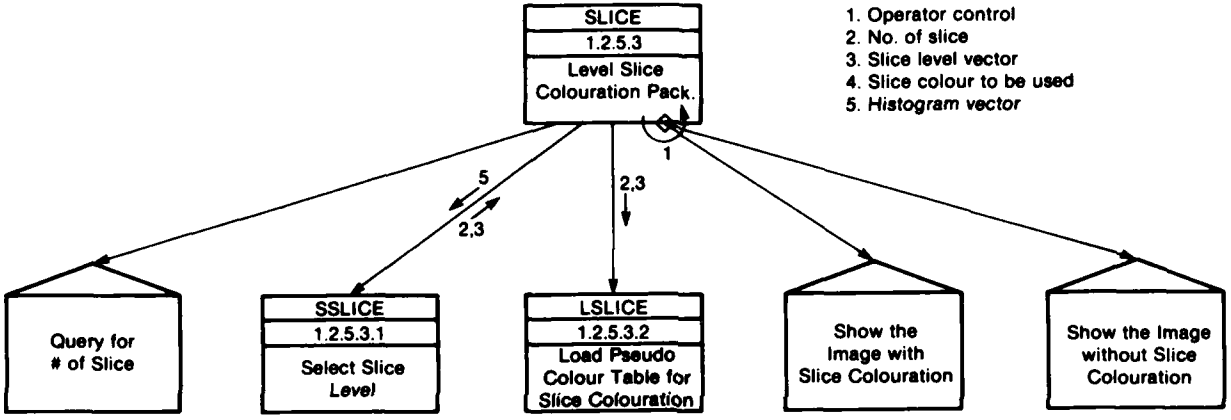
3.1.35 Structure of the DDUMP procedure



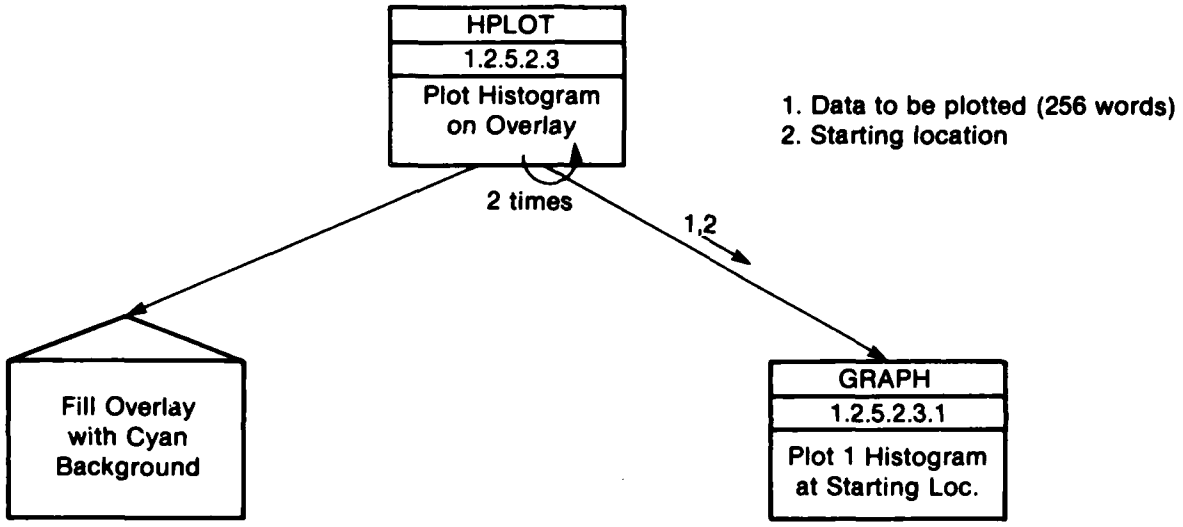
3.1.34 Structure of the ZOOM procedure



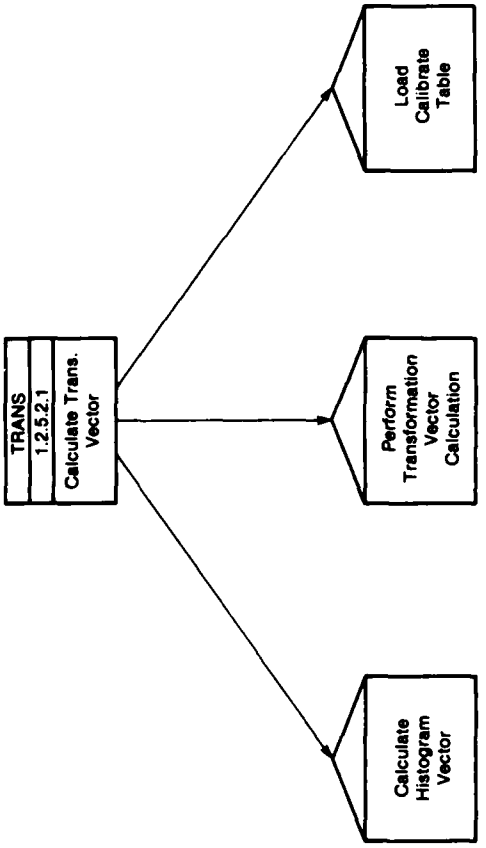
3.1.33 Structure of the SSLICE procedure



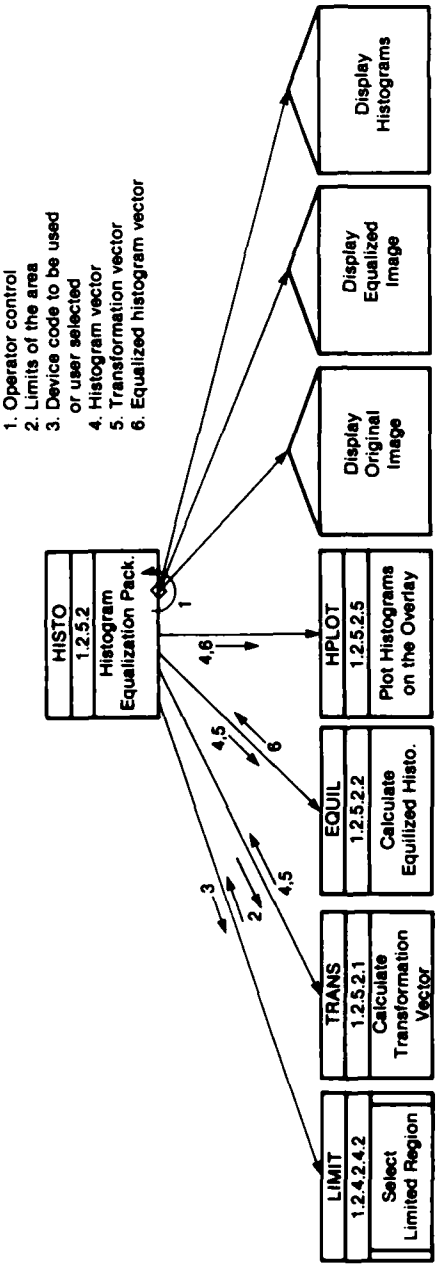
3.1.32 Structure of the SLICE procedure



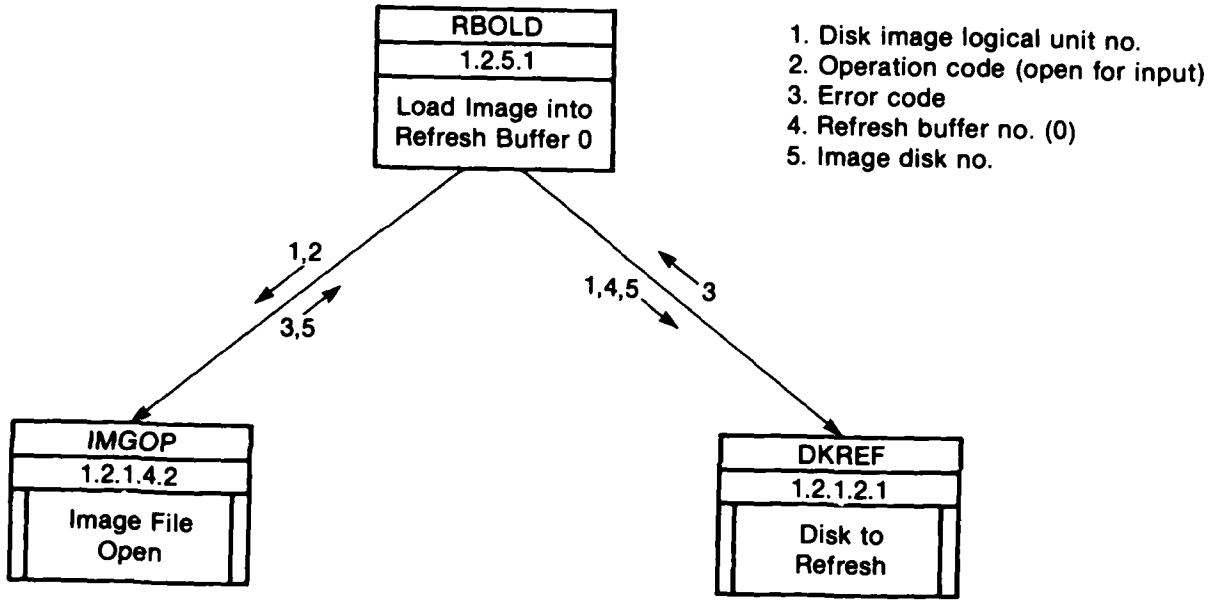
3.1.31 Structure of the HPLOT procedure



3.1.30 Structure of the TRANS procedure



3.1.29 Structure of the HISTO procedure



3.1.28 Structure of the RBOLD procedure

TABLE 6

OPERATION DESCRIPTION PORTION OF LOG FILE RECORD FOR OPERATION CODE 0 (ERIM TAPE TO DISK)

WORD	VARIABLE DESCRIPTION	VARIABLE TYPE
165	Start line Number	Integer
166	Line Increment	Integer
167	Start Pixel Number	Integer
168	Pixel Increment	Integer
169	Number of Pixels per line	Integer
170	Number of lines	Integer
171-256	Spare	---

TABLE 7

OPERATION DESCRIPTION FOR OPERATION CODE 1 (IMAGE CREATED BY RATIO)

WORD	VARIABLE DESCRIPTION	VARIABLE TYPE
165-166	Shift Value	Real
167-168	Multiplier Value	Real
169-170	Power Value	Real

TABLE 8

OPERATION DESCRIPTION FOR OPERATION CODE 2 (IMAGE CREATED BY CLASS)

WORD	VARIABLE DESCRIPTION	VARIABLE TYPE
165-166	Confidence Level Used	Real
167-172	Up to 6 element Signature	Integer
173-178	Error Vector (6 elements)	Integer
179-256	Spare	---

TABLE 9

OPERATION DESCRIPTION FOR OPERATION CODE 3
(IMAGE CREATED BY RRFDK)

WORD	VARIABLE DESCRIPTION	VARIABLE TYPE
165-256	Spare	---

TABLE 10

OPERATION DESCRIPTION FOR OPERATION CODE 4
(IMAGE CREATED BY ALGCOMB)

WORD	VARIABLE DESCRIPTION	VARIABLE TYPE
165-244	Equation terminated by a null " 0 "	ASCII
245-256	Spare	---

TABLE 11

MESSAGE FILE IMAGER.ER RECORD FORMAT

BYTE	DESCRIPTION	TYPE
1	Number of Integer arguments (0-9)	ASCII
2-256	Fortran format specifications of the message	ASCII

3.3 MODULE SPECIFICATIONS

3.3.1 IMAGER (1.0) Mainline for Imager Task

Function

This module is the mainline for the IMAGER task. It asks the operator if initialization is required, after which control is given to the module MENU.

Inputs

The only input to this module is the operator response as to whether initialization is desired.

Processing

If initialization is required, this module calls module INITIAL. IMAGER subsequently calls module MENU and upon returning, outputs a closing message to the console and returns control to RDOS.

Outputs

The output of this module consist of an opening message to the console when called and a closing message to the console when completed. Between these two messages, a prompt is sent to the console questioning the operator as to whether initialization is desired.

Error Handling

Any errors detected in the operator input are ignored.

3.3.2 INITIAL (1.1) Perform Initialization for IMAGER

Function

The function of this module is to initialize the NORPAK display buffers and to reset the interrupts.

Inputs

None.

Processing

This module calls the NORPAK library routine INITR35 which initializes the NORPAK display buffers and other NORPAK related functions. This module then deletes the refresh headers located on disk.

Outputs

None.

Error Handling

None.

Calling Sequence

Call INITIAL.

3.3.3 MENU (1.2) Imager Main Menu

Input Name(s) and Specification

None.

Module Function(s)

This module offers the following menu to the operator:

- Route Images (RI)
- System Parameter List and Update (SP)
- Multi-Spectral Analysis Package (MS)
- Mono-Spectral Analysis Package (SS)
- Exit (EX)

Upon user selection call the appropriate module

For RI call RTIMG
For SP call SPLU
For MS call MSA
For SS call SSA
For EX return

Error Handling

Ignore any operator input that doesn't match one of the menu choices.

Calling Sequence

Call MENU.

3.3.4 RTIMG (1.2.1) Route Image

Input Name(s) and Specification

None.

Output Name(s) and Specification

None.

Module Function(s)

This module presents to the operator the following menu:

- Route Tape to Disk (TD)
- Route Disk to Refresh (DR)
- Route Disk to Tape (DT)
- Route Refresh to Disk (RD)
- Route Refresh to Monitor (RM)
- Return to Main Menu (RT)

Upon operator selection call the appropriate module

For TD call RTPDK
For DR call RDKRF
For DT call RDKTP
For RD call RRFDK
For RM call RRFMO
For RT return

Error Handling

Ignore any operator input that doesn't match one of the menu choices.

Calling Sequence

CALL RTIMG

3.3.5 RTPDK (1.2.1.1) Route Tape Image to Disk

Function

The function of this module is to mount a tape for reading and determine the format of the tape.

Inputs

Inputs from the calling routine include the disk and tape channel numbers, the number of lines to read and the number of pixels per line. Other inputs to this module are two character code indicating the tape format (Input by the operator) and an error code from TPMNT.

Processing

Initially, the tape is mounted by calling the module TPMNT. Then the module corresponding to the tape format is given control. These modules are RDERM (read ERIM format tape) and RDSTD (read Standard format tape). Upon return from RDERM or RDSTD modules, control is returned to the calling routine.

Outputs

This module outputs a list of available tape formats and an operator prompt.

Error Handling

If the error code returned from module TPMNT indicates an error, control is returned to the calling routine. If an input error is encountered in response to the operator prompt, the list of formats and operator prompt are repeated.

Calling Sequence

Call RTPDK (DUN, MTU, NLIN, NPIX) where all parameters are input.

DUN:	disk channel number	(Integer)
MTU:	tape channel number	(Integer)
NLIN:	Number of lines to read	(Integer)
NPIX:	Number of pixels per line	(Integer)

3.3.6 TPMNT (1.2.1.1.1) Mount a Tape

Function

The function of this module is to determine if opening and/or initialization should be performed on a given tape.

Inputs

Input from the calling procedure is the magnetic tape channel number. Other inputs to this module consist of a carriage return to indicate that the desired tape has been mounted on the drive and an error code returned from the module TPINI.

Processing

Upon input of a carriage return from the operator, the tape status is checked and if ready, module TPINI is called to initialize and open the unit. If TPINI returns an error, then the operator is again prompted to enter a carriage return when a tape has been mounted. This procedure is repeated up to four times after which an error is returned to the calling routine if unsuccessful. If successful, a no error condition is returned to the calling routine.

Outputs

The outputs of this module are an operator prompt displayed on the console and an error code returned to the calling routine.

Error Handling

Any operator inputs followed by a carriage return are acceptable. Therefore there is no error handling.

Calling Sequence

Call TPMNT (MTU, IRING, ERR).

MTU:	Magnetic tape logical unit number	(Integer)	(Input)
IRING:	Write ring present/absent	(Integer)	(Output)
ERR:	Operation successful/unsuccessful	(Integer)	(Output)

3.3.7 TPINI (1.2.1.1.1.1) Initialize a Tape

Function

The function of TPINI is to initialize and/or open the magnetic tape unit.

Inputs

The sole input is the tape unit number.

Processing

The tape unit is initialized and opened.

Outputs

The output of this module is an error code with is sent to the calling routine. An error code of one indicates success. If unsuccessful an error message is output to the line printer.

Error Handling

If an error is encountered while initializing or opening the tape unit, the corresponding error code is returned to the calling routine and an error message displaying the error code is output to the line printer.

Calling Sequence

Call TPINI (MTU, ERR).

MTU:	Magnetic tape logical unit number	(Integer)	(Input)
ERR:	Return error status	(Integer)	(Output)

3.3.8 RDERM (1.2.1.1.2) Read ERIM Tape

Function

The function of RDERM is to transfer images from ERIM format tapes to Standard format disk files.

Inputs

Inputs from the calling routine include the unit numbers for the tape and disk, the number of lines per image and the number of pixels per line. Input from module TPDAT is the header. The module GETNM inputs the disk file name and module RDTP inputs the tape data and an error code. Modules SPREC and ODKWT also return an error code.

Processing

Initially, the tape characteristics are obtained by calling the module TPDAT. Then a call to module ERMIN inputs the file transfer specifications from the operator. The tape is then positioned by module SPREC and the appropriate disk file is opened by module ODKWT. SEQNO is called to obtain the next log file sequence number. An image is transferred one line at a time using module RDTP and the FORTRAN WRITE BINARY (to disk) command. After transferral of each image, the disk file is closed.

Outputs

The prime outputs are the disk files containing the images. Additional outputs include informative messages displayed on the console. These are displayed while module TPDAT is getting the tape characteristics and during the transfer of each image. Error diagnostics are also output to the console.

Error Handling

If an error is returned by module TPDAT, control is returned to the calling module. An error indication returned from module SPREC or ODKWT, a disk write error or a tape read error (other than a parity error) will result in the current disk file being deleted, remaining image transfers oborted, a diagnostic message on the line printer and return of control to the calling routine. In addition, if a disk write error occurs, it is reported on the line printer.

Calling Sequence

Call RDERM (DUN, MTU, NLIN, NPIX) where all parameters are input.

DUN:	Disk logical unit number	(Integer)
MTU:	Magnetic tape logical unit number	(Integer)
NLIN:	Number of lines to read	(Integer)
NPIX:	Number of pixels/line	(Integer)

3.3.9 TPDAT (1.2.1.1.2.1) Get Header Info From Tape

Function

The function of this module is to get the characteristics of an ERIM format tape.

Inputs

The prime input is the data on tape. Input from the calling routine is the tape unit number.

Processing

Initially, the tape is rewound and the header is read by module RDTP. The number of bands and the channel numbers are determined by reading consecutive records until the first channel number is repeated (see ERIM format Appendix A). The Standard format header is then constructed from information contained in the tape header. Control is then returned to the calling routine.

Outputs

Output from this module are a Standard format header and an error code.

Error Handling

If a tape read error occurs on the header, the ASCII data is blanked and the numeric data is zeroed. If a tape read error occurs while the channel numbers are being determined, the module will continue to restart this procedure until a total of 100 records have been read. If an error occurs after this, an error is returned in the error code.

Calling Sequence

Call TPDAT (MTU,HDR, ERR).

MTU:	Tape channel number	(Integer)	(Input)
HDR(160):	ERIM header	(Integer)	(Output)
ERR:	Return error status	(Integer)	(Output)

.3.10 RDTP (1.2.1.1.2.5) Read a Tape Record

unction

The function of this module is to read a record from magnetic ape.

nputs

Input from the calling routine are the tape unit number, record ength and buffer address. Input from the tape unit is the tape data.

rocessing

This module reads a record of data from tape to the input buffer nd returns control to the calling routine with a return code indicating uccess.

lutputs

Output to the calling module are the tape data, a return code and he number of words read.

rror Handling

If an error occurs while reading from tape, it is reported to the line printer and control is returned to the calling routine with the return code set to indicate the nature of the error.

alling Sequence

Call RDTP (MTU, NWD, DATA, NWDR, IER).

MTU:	Magnetic tape logical unit number	(Integer)	(Input)
NWD:	Number of words to read	(Integer)	(Input)
DATA:	Input buffer	(Integer)	(Input)
NWDR:	Number of words read	(Integer)	(Output)
IER:	Return error status	(Integer)	(Output)

.11 ERMIN (1.2.1.1.2.2) Get Data Transfer Specs from ERIM Tape

ction

The function of module ERMIN is to obtain the image transfer cifications from the operator for transferring images from ERIM format e.

uts

Inputs from the calling module are the disk unit number and the ber of pixels per line. The inputs required from the operator are the d numbers, the start line number, the line increment and the destination k file for each image transfer. Input from module RPROT is a return code icating whether a disk file is write protected.

cessing

For each image transfer, the operator is queried for the band ber, the start line number, the line increment and the destination disk e number. A zero entered for the band number indicates the end of rator inputs. The band number is checked against those on the tape. If re is no match the query is repeated. The destination disk file is cked first to determine if it is write protected. Ii it is, the operator l be asked if overwriting of the file is desired. If this is not ired, the destination disk file number is asked for again. Secondly, the k file number is checked to determine if it is already used as the tination for a previous transfer. If this is true, it is reported to the rator and the query for a destination disk file is repeated. The rator is asked to verify the specifications of each image transfer and the option of aborting at this point. If the operator wishes to abort, trol is returned to the calling routine with the number of transfers al to zero. If the operator does not verify the specifications for an ge transfer, those specifications are deleted and the operator must input ew set.

When operator input is complete, the image transfers are sorted in ending start line number order to minimize future tape movement. Control then returned to the calling routine.

puts

None.

.21 RDKTP (1.2.1.3) Route Disk Image to Tape

ction

The function of this module is to transfer images from disk to ndard format tape.

uts

Input by the operator are the source disk file numbers and if nsferring images to a new tape, a tape number. Input from module TPMNT a write ring indicator and an error return. The tape label and headers well as an error code are input from module RDTP. Error codes are also urned by modules STDTST, ODKRD, SPFIL, WEOF and SEFB.

cessing

Initially the tape is mounted by calling module TPMNT. If the te ring return indicates there is no write ring, it is reported on the e printer and control is returned to the calling routine. The tape is n checked for the proper label by calling module STDTST. If it is a new e, the operator is queried for a four digit tape number and a label is tten to tape. If the tape label is not proper, control is returned to calling routine. The tape is then positioned to the end of the last e noting the number of files on tape by calls to modules SPFIL and RDTP.

The operator is then queried for the source disk file numbers. A o entry indicates the end of operator input. The disk file is checked to ure that it exists. If it does not, it is reported to the console and query is repeated. If the number is out of range or has been repeated, query is also repeated.

At the end of the operator input, the files are transferred one at ime from disk to tape with the completion of each transfer being reported the line printer. At the completion of all transfers, a double end of e is written to tape by calling module WEOF and control is returned to calling routine.

puts

Output to the operator are the queries for the source disk file ber and the new tape number. The prime outputs are the image files and el to the magnetic tape. Output to called modules TPMNT, STDTST, SPFIL, F, SEFB and RDTP is the tape unit number, the disk unit number and file ber to ODKRD, the record length and buffer address to RDTP and the number files to skip to SPFIL.

3.20 DKREF (1.2.1.2.1) Perform Disk to Refresh Transfer

put Name(s) and Specification

DKLUN: Disk image file logical unit no.

REFNO: Refresh buffer no.

DKINO: Image file no.

tput Name(s) and Specification

ERCOD: Error code

1 Successful
2 Process stopped by an EOF
3 Any other disk error.

dule Function(s)

This module will copy the disk file header to the proper disk file d then transfer the image data to the specified refresh buffer.

ror Handling

If a disk error occurs, report it on the line printer and transfer ntrol to the calling routine with ERCOD set to the proper value.

lling Sequence

Call DKREF (DKLUN, REFNO, DKINO, ERCOD).

Calling Sequence

Call RDKRF (DUN, NLIN, NPIX) where all parameters are input.

DUN: disk logical unit number (Integer)
NLIN: Number of lines to read (Integer)
NPIX: Number of pixels per line (Integer)

3.3.19 RDKRF (1.2.1.2) Route Disk Image to Refresh Buffer

Function

The function of module RDKRF is to move images from disk files to the NORPAK refresh buffers.

Inputs

Input from the operator are the source disk file numbers and the destination refresh bank numbers. Input from the disk are the image files. Module UNPACK inputs an unpacked image line. Modules ODKWT and ODKRD return an error code.

Processing

The operator is asked for the source disk file number, destination disk file number and verification for each image transfer. A zero input as the source file indicates end of operator input. When asked to verify each image transfer, the operator may elect to accept it, change it or immediately return to the calling module. The range of both source and destination file numbers is checked. If either is out of range the query is repeated.

The source file is opened by module ODKRD and the disk to refresh transfer is performed by DKREF, for each image to be transferred. The display is also set to the refresh bank being loaded so that the image can be seen. All disk files are then closed and control is returned to the calling routine.

Outputs

The prime outputs are the images in the refresh banks and the refresh bank headers on disk. Also output are the queries to the operator, the disk unit number and file numbers to module ODKRD, and a return error status from DKREF.

Error Handling

Operator input errors are handled by repeating the query. Disk read errors are handled by blanking the ASCII data, zeroing the numeric data and reporting the nature and position of the error to the line printer. Disk write errors and disk file opening errors are reported on the line printer and cause pending data transfers to be aborted and control returned to the calling routine.

Calling Sequence

Call STDTST (MTU, TNO, IER).

MTU:	Magnetic tape logical unit number	(Integer)	(Input)
TNO:	Tape number	(Integer)	(Output)
IER:	Return error status	(Integer)	(Output)

3.3.18 STDTST (1.2.1.1.3.1) Determine if Tape is Standard

Function

This module verifies that the tape mounted on the tape drive is Standard format with the correct tape number.

Inputs

The calling routine inputs the tape unit number. The operator inputs the tape number. Module RDTP inputs tape data and an error code. Modules REW and SEFF also return an error code.

Processing

Initially, the tape is rewound by module REW. The tape header is then input by module RDTP. If the tape is not Standard format, it is reported on the line printer and control is returned to the calling routine with the error code set to indicate an improper format. If the tape is new (double EOF at beginning of tape), control is returned to the calling routine with the error code set to indicate this fact. Otherwise, the operator is prompted for the tape number. If the tape numbers do not match, the operator is given a second chance and if still no match, control is returned to the calling routine with the error code set to indicate a wrong tape. If the tape number match, the tape is brought to the beginning of the first image file by module SEFF and control is returned to the calling routine with the error code set to indicate success.

Outputs

The prime output is the error code returned to the calling routine. The tape unit number and the number of files to skip are output to SEFF. The tape unit number, record length and buffer address are output to module RDTP.

Error Handling

All inputs from the operator are considered valid tape numbers. If a tape error occurs, control is returned to the calling routine witht he appropriate error code.

Error Handling

In the event of an operator input error, the prompt is repeated. If a tape read error or a disk write error occurs, it is reported to the line printer, the current disk file is deleted, all pending data transfers are aborted and an informative message is sent to the line printer after which control is returned to the calling routine.

Calling Sequence

Call RDSTD (DUN, MTU, NLIN, NPIX) where all parameters are input.

DUN:	disk logical unit number	(Integer)
MTU:	Magnetic tape logical unit number	(Integer)
NLIN:	Number of lines to read	(Integer)
NPIX:	Number of pixels/line	(Integer)

3.3.17 RDSTD (1.2.1.1.3) Read a Standard Tape

Function

Module RDSTD transfers images from Standard format tape to disk files.

Inputs

Input from the calling routine are the unit numbers for the tape and disk drives, the number of lines per image, the number of pixels per line and the address of the line buffer. Input from the operator are the image transfer specifications which include the source tape file number, the destination disk file number and verification for each image to be transferred. Module RDTP inputs tape data along with an error code. RPROT input tells wether or not a disk file is write protected and GETNM inputs the disk file name. An error code is input on return from modules SPFIL, ODKWT and STDTST.

Processing

The first step is to check whether or not the tape is properly formatted by calling module STDTST. If not, control is returned to the calling module. The operator is then queried for the source and destination file numbers. A zero entered for the source file number indicates the end of operator input. The destination disk file is checked first to see if it is in the proper range, next to see if it is duplicated and finally a call to module RPROT checks if it is write protected. If it is write protected the operator is asked if overwriting of the file is desired. If the operator does not wish to overwrite the file or if the file number is duplicated, the destination file number query is repeated.

At the end of operator input, the image transfers are sorted in ascending order of source file number to minimize tape movement during transfer. The tape is then positioned by module SPFIL and the appropriate disk file is opened by module ODKWT. Each record is then input by module RDTP and written to disk for each image (file) transfer. The completion of each image transfer is reported to the line printer. When all transfers are complete, control is returned to the calling routine.

Outputs

The prime outputs are the disk files containing the transferred images. Also output are the tape unit number to module STSTST, the disk unit and file number to modules RPROT and ODKWT, the number of files to space to module SPFIL and the tape unit number, record length and buffer address to module RDTP.

3.3.16 SEQNO (1.2.1.1.2.4) Get Next Sequence Number

Input Name(s) and Specification

None.

Output Name(s) and Specification

LOGNO: Index to next log entry (Integer)

RECNO: Number of entries in log file + 1 (Integer)

FRSTNO: Sequence number of first log entry (Integer)

ERCOD: Error code (Integer)

1 = Operation was Successful
2 = Operation was Unsuccessful

Module Function(s)

a) Open IMAGER.LO, the log file.

b) Read LOGNO, RECNO and FRSTNO from the first record.

c) Increment LOGNO and RECNO.

d) Write these values back to first record of IMAGER.LO.

e) Close IMAGER.LO.

Error Handling

If a disk error occurs during the execution of this module inform the operator of the error type, close IMAGER.LO and return to the calling module with ERCOD = 2.

Calling Sequence

Call SEQNO (LOGNO, RECNO, FRSTNO, ERCOD).

3.3.15 ODKWT (1.2.1.1.2.3) Open Disk File for Write

Function

The function of this routine is to open a disk file for the purpose of writing.

Inputs

Input from the calling routine are the disk unit number, disk file number and the record length. Input from module GETNM is the disk file name.

Processing

Initially, the disk file name is obtained from module GETNM. The file is then opened by a system call. If it exists, the file is closed, deleted, created and opened once again. Then control is returned to the calling routine with a successful return code. If the disk file does not originally exist, it is created, opened and return is made to the calling routine with a return code indicating success.

Outputs

Output to the module GETNM is the disk file number. Output to the calling routine is a return code.

Error Handling

If an error occurs while opening, creating or deleting a disk file, the error is reported to the line printer and control is returned to the calling routine with a return code which indicates the operation was unsuccessful.

Calling Sequence

Call ODKWT (DUN, DFN, NPIX, ERR).

DUN:	disk logical unit number	(Integer)	(Input)
DFN:	disk file number	(Integer)	(Input)
NPIX:	Number of pixels/line	(Integer)	(Input)
ERR:	Return error status	(Integer)	(Output)

3.3.14 GETNM (1.2.1.1.2.6) Get Imager File Name

Function

The function of this module is to get a disk file name, given the disk file number.

Inputs

Input from the calling routine is the disk file number.

Processing

The disk file number is used in a look-up table to obtain the corresponding file name. Control is then returned to the calling routine.

Outputs

Output to the calling routine is the disk filename.

Error Handling

None.

Calling Sequence

Call GETNM (DFN, FNAME).

DFN:	disk file number	(Integer)	(Input)
FNAME (7):	Filename	(Integer)	(Output)

3.3.13 ODKRD (1.2.1.1.2.2.1.1) Open Disk File for Read

Function

The function of this routine is to open a disk file for the purpose of reading.

Inputs

Input from the calling routine are the disk unit number, disk file number and the record length. Input from module GETNM is the disk file name.

Processing

The name of the disk file is obtained by a call to module GETNM. The corresponding disk file is then opened with a system call. If the file opening is successful, control is returned to the calling routine with a return code indicating suscess. If the disk file does not exist, control is returned to the calling routine with a return code indicating a non-existent file.

Outputs

Output to module GETNM is the disk file number. Output to the calling routine is a return code.

Error Handling

If an error occurs while attempting to open this disk file, it is reported on the line printer and control is returned to the calling routine with the appropriate return code.

Calling Sequence

Call ODKRD (DUN, DFN, NPIX, ERR).

DUN:	disk logical unit number	(Integer)	(Input)
DFN:	disk file number	(Integer)	(Input)
NPIX:	Number of pixels/line	(Integer)	(Input)
ERR:	Return error status	(Integer)	(Output)

3.3.12 RPROT (1.2.1.1.2.2.1) Check Protect Status of Disk File

Function

The function of this module is to check if a disk file is write protected.

Inputs

Input from the calling routine are the disk unit number, disk file number and the record size. Input from the disk file is the protect word which is the first word in the header. Module ODKRD returns an error code.

Processing

The disk file is opened by a call to module ODKRD. If the error code returned indicates that the file does not exist, the protect word to be returned to the calling return is cleared. If it exists, the protect word is read and the disk file is closed. If there is an error in the opening of the disk file, the protect word is set. Control is returned to the calling routine.

Outputs

The output consists solely of the protect word returned to the calling routine. If it is set, the disk file is write protected.

Error Handling

None.

Calling Sequence

Call RPROT (DUN, DFN, NPIX, IPRT).

DUN:	disk logical unit number	(Integer)	(Input)
DFN:	disk file number	(Integer)	(Input)
NPIX:	number of pixels/line	(Integer)	(Input)
IPRT:	protect status word	(Integer)	(Output)

Error Handling

If an operator input error is encountered, the operator is first notified of the error and then the prompt is repeated. An entry which is out of range is also treated as an operator input error.

Calling Sequence

Call ERMIN (NPIX, DUN) where both parameters are input.

NPIX: Number of pixels per line
DUN: Disk logical unit number

Error Handling

An operator input error is handled by repeating the query. A tape writing or spacing error and a disk opening or read error are reported to the line printer and result in a double end of file being written after the last complete tape file after which control is returned to the calling routine.

Calling Sequence

Call RDKTP (DUN, MTU, NLIN, NPIX) where all parameters are input.

DUN: disk logical unit number
MTU: magnetic tape logical unit number
NLIN: number of lines to read
NPIX: number of pixels per line

3.3.22 RRFDK (1.2.1.4) Refresh Buffer to Disk

Input Name(s) and Specification

None

Output Name(s) and Specification

None.

Module Function(s)

a) Get the Refresh Buffer to be copied (0 to 3) and the disk image file to be created (IMGOP)

b) Create the image file header identify the operation as Refresh to Disk copy code = 3. Number of parents = 0 (COPH)

c) Copy the Refresh Buffer to the specified Image File

d) Repeat until a negative Refresh Buffer is specified.

Error Handling

Repeat every operator's prompt until an in range answer is given.

If any disk error occurs inform the operator, close the open file, delete it, and return to the calling routine.

Calling Sequence

Call RRFDK.

3.3.23 EDIT (1.2.1.4.1) Edit the Comment Portion of a Disk File Header

Function

This module replaces the comment portion of the header of a disk file with operator input.

Inputs

Input from the calling routine is the header data. Input from the operator are the new comments which are up to 100 characters in length.

Processing

The module prompts the operator to input up to 100 characters of comments to replace the previous comments. These are then returned to the calling routine.

Outputs

The prime output is the new header data which is sent to the calling routine. Also output is the operator prompt.

Error Handling

If an error is encountered in writing to disk, it is reported to the line printer and control is returned to the calling routine.

Calling Sequence

Call EDIT (HDR) where

HDR(160): is the header portion of image.

3.3.24 IMGOP (1.2.1.4.2) Open an Image

Input Name(s) and Specification

CHNL: Disk file logical unit number (Integer)

PC: Process code (Integer)

± 1 Open a file for read
± 2 Open a file for write

No query for image number if negative

IMGNO: This is an input parameter if PC = -1 or -2. In this case, IMGNO is an image file number passed by the calling routine.

Output Name(s) and Specification

IMGNO: This is an output parameter if PC = +1 or +2. In this case, IMGOP will prompt operation for file number. (Integer)

RC: Return code

1 = File was opened (Integer)
2 = File not opened

Module Function(s)

(i) For PC > 0

a) Ask the operator for disk file number, the acceptable range is 1 to 16.

b) If the image file number = 0, exit with RC = 2.

c) If PC = 2, check output image for write protection status.

d) If write protected, ask permission to overwrite if no return to a).

e) Open the file. Call ODKRD if file is to be opened for read. Call ODKWT if file is to be opened for write.

f) Check if the file was opened properly. If an error occurred, go to a).

(ii) For PC <0

a) Check Disk file number for range 1-16, exit with RC = 2 if not in range.

b) If PC = -2 delete old file if exists and create a new one. Set RC = 2 if any error and exit.

c) Open the file (ODKRD or ODKWT).

d) Check if the file opened properly. If an error set RC = 2 and exit.

Error Handling

Report any disk error on the line printer and Set RC = 2.

Calling Sequence

Call IMGOP (CHNL, IMGNO, PC, RC).

3.3.25 RRFMO (1.2.1.5) Route Refresh To Monitor

Input Name(s) and Specification

None.

Output Name(s) and Specification

None.

Module Function(s)

a) Query the operator for the Refresh Buffer number to be used (0-3).

b) Route the Refresh to Display.

c) Repeat steps a and b until a negative Refresh Buffer is specified.

Error Handling

Requery the operator until a Refresh Buffer from 0-3 or a negative number is entered.

Calling Sequence

Call RRFMO.

3.3.26 SPLU (1.2.2) System Parameters List & Update

Input Name(s) and Specification

None.

Output Names(s) and Specification

None.

Module Function(s)

This module offers the following menu to the operator.

- Tape Header List (TL)
- Disk Header List & Update (DL)
- Refresh Header List (RL)
- *Log File Output* (LO)
- Return to Main Menu (RT)

upon operator selection call the appropriate module.

For TL call TLIST
For DL call DLIST
For RL call RLIST
For LO call LOGLS
For RT return

Error Handling

Ignore any operator input that doesn't match one of the menu choice.

Calling Sequence

Call SPLU.

3.3.27 DLIST (1.2.2.1) List Disk Image Parameters

Function

The function of this module is to determine the number of the disk file from which the operator wants to list the header and allow the operator to replace the comment section of the header.

Inputs

Input from the operator are the disk file number and a decision as to whether editing of the user comment is desired. Input from the calling routine is the disk unit number and line printer flag. Input from module ODKRD is a return code. The disk file header data is input from disk.

Processing

The module prompts the operator for a disk file number. If the number is negative, control is returned to the calling routine. If the number is out of range, the operator is notified and the prompt is repeated. After a file number has been accepted, the corresponding disk file is opened by a call to module ODKRD and the header data is read. The header is then listed by a call to module LIST. The operator is then queried as to whether it is desired to replace the comment section of the header. If yes, the replacement is accomplished by a call to module EDIT. The disk file is then closed and the operator is prompted for another disk file number with the above procedure repeated.

Output

Output to the console are the operator prompts. Output to module ODKRD are the disk unit and file numbers and the record length. Output to module LIST are the header data and line printer flag. Output to module EDIT is the header data.

Error Handling

If an operator input error is encountered, it is reported on the console and the initiating prompt is repeated. If a disk error occurs, it is reported on the console and the operator is prompted for a new disk file number.

Calling Sequence

Call DLIST (DUN, LP) where both parameters are input.

DUN: disk logical unit number (Integer)
LP: line printer flag (Integer)

3.3.28 LIST (1.2.2.1.1) List Header Information

Function

The function of this module is to list header data on the console and/or the line printer.

Inputs

Input from the calling routine are the header data and the line printer flag.

Processing

The header data is listed on the console. If the line printer flag is set, the header data is also listed on the line printer. Control is then returned to the calling routine.

Outputs

The output consists of header listings on the console and/or line printer.

Error Handling

None.

Calling Sequence

Call LIST (HDR, LP) where both are input parameters.

HDR(160): is the header of image file
LP: is the line printer flag

.3.29 RLIST (1.2.2.2) List Refresh Image Parameters

ınction

The function of this module is to determine the number of the efresh bank for which the operator wishes to list the header.

ıputs

Input from the operator is the refresh bank number. Input from ıe calling routine are the disk unit number and line printer flag. Input rom module ODKRD is a return code. The disk file header data is input from isk.

rocessing

The operator is prompted for the number of a refresh bank. If the ımber entered is negative, control is returned to the calling routine. If he number is out of range, the operator is notified and the prompt epeated. After a refresh bank number has been accepted, the corresponding isk file is opened by module ODKRD and the header data is read. This data s then listed by a call to module LIST. The disk file is then closed and he operator is prompted for the number of another refresh bank with the bove procedure repeated.

utputs

Output to the console is the operator prompt. Output to module DKRD are the disk unit and file numbers and the record length. Output to odule LIST are the header data and line printer flag.

rror Handling

An operator input error is reported on the console and the prompt s repeated. A disk error will be reported on the console and the operator ill be prompted for another refresh bank number.

alling Sequence

Call RLIST (RUN, LP) where both parameters are input.

RUN: refresh buffer number (Integer)
LP: line printer flag (Integer)

3.30 TLIST (1.2.2.3) List Tape Image Parameters

ıction

The function of this module is to determine the tape files for ich the operator wishes to list the headers.

ɔuts

Input from the calling routine are the tape unit number and the ıe printer flag. Input from the operator are the first and last file ıbers for which to list the headers. Input from modules TPMNT, STDTST and FIL are return codes. Input from module RDTP are the tape header data and return code.

ɔcessing

Initially, the tape is mounted by a call to TPMNT. Then the tape rmat and number are verified by a call to STDTST. The operator is then eried for the first and last file numbers for which the headers are to be sted. A zero entered as the first file will cause control to return to e calling routine. Values out of range will cause the operator to be tified and the prompt repeated. The tape is then positioned to the first le whose header is to be listed by a call to SPFIL. The headers of the ecified files are then listed by repeated calls to LIST and SPFIL. If the ıe printer flag is not set, a pause will occur after each header. Control returned to the calling routine after all specified file headers have en listed and the tape rewound.

tputs

The prime outputs are the header data to module LIST along with e line printer flag. Output to the console is the operator prompt. tput to modules TPMNT and STDTST is the tape unit number. Output to SPFIL e the number of files to skip and the tape unit number. Output to module TP are the record length, tape unit number, and input buffer address.

ror Handling

In the event of an operator input error, it is reported on the nsole and the prompt is repeated. If a tape error occurs, other than a ad error, control is returned to the calling routine.

ling Sequence

Call TLIST (MTU, LP) where both parameters are input.

MTU: magnetic tape logical unit number (Integer)
LP: line printer flag (Integer)

31 LOGLS (1.2.2.4) List Log File

: Name(s) and Specification

None.

it Name(s) and Specification

None.

le Function(s)

a) Open IMAGER.LO, the IMAGER log file.

b) Read header record of IMAGER.LO to get NXTSEQ (next available sequence number) and FRSTSEQ (first sequence number).

c) Display the following menu:

Print the entire log	(PL)
Print a portion of the log	(PP)
Create a new log file	(CL)
Return to SPLU menu	(RT)

Display first available sequence number (FRSTSEQ) and last available sequence number (NXTSEQ - 1).

d) Read operator selection. If PP selected, ask for first and last sequence numbers desired. Validate that they are in range and that first sequence number < = last sequence number.

e) If PL or PP desired, print on the line printer, one image per page, a formatted version of the information in the log. See Section 3.2.3 for descriptions of each type of log entry.

f) If CL desired, close IMAGER.LO and delete it. Create it again and open it. Set FRSTSEQ = NXTSEQ, and NXTREC = 2.

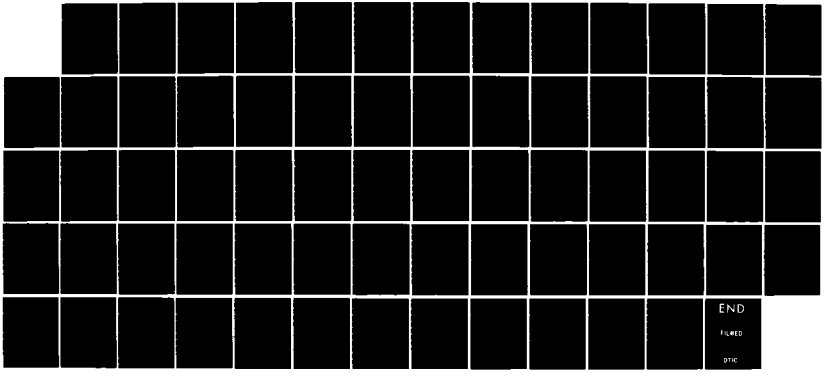
g) If RT selected, close IMAGER.LO and exit or else repeat steps (c)-(f).

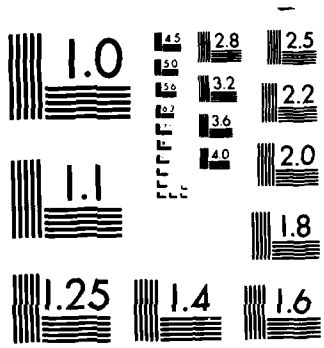
r Handling

If a disk error occurs inform the operator of the error, close the file and return to the calling module.

AD-A151 752 DESIGN SPECIFICATIONS FOR IMAGER: THE DREO (DEFENCE RESEARCH ESTABLISHMEN..(U) DEFENCE RESEARCH ESTABLISHMENT OTTAWA (ONTARIO) C BROCHU ET AL. SEP 83 2/2

UNCLASSIFIED DREO-TN-82-15 F/G 9/2 NL





If operator selects an invalid range when queried for first and last sequence numbers desired, print an error message and re-query.

Calling Sequence

Call LOGLS.

3.3.32 to 3.3.67

Reserved for future use.

3.3.68 MSA (1.2.4) Perform Multi-spectral Analysis Processing

Input Name(s) and Specification

None.

Output Name(s) and Specification

None.

Module Function(s)

This module will present to the operator the following menu:

- Ratio Image (RATIO)
- Intensity Correlation (INCOR)
- Target Classifier (CLASS)
- Colour Composite Image (COLCO)
- Algebraic Combination of Images (ALGCOMB)
- Return.

Error Handling

None.

Calling Sequence

Call MSA.

3.3.69 RATIO (1.2.4.1) Ratio of Two Images

Input Name(s) and Specification

None.

Output Name(s) and Specification

None.

Module Function(s)

a) Get from the operator 3 disk file numbers (2 for input, 1 for output) and the transformation coefficients (RATIN).

(b) Call COPH to create the output file header and log entry.

(c) Perform the ratio calculation on the 2 input images for each line (RDIV).

(d) Store the result in the output file.

(e) Close all open files.

Error Handling

If any disk error occurs, during the execution of this module report the error on the line printer and return to the caller. Close all open files and delete the output file if empty.

Calling Sequence

Call RATIO.

3.3.70 RATIN (1.2.4.1.1) Get Image File Numbers for Ratio

Input Name(s) and Specification

None.

Output Name(s) and Specification

LUN(3): Disk Image Logical Unit No.

1st is output
2nd is numerator (NUM)
3rd is denominator (DENUM)

SHFT: Bias on the output image (real).

MLTPLR: Gain applied to the ratio value (real).

PWR: Exponent for DENUM pixel value (real).

RC: Return Code

1 success
2 failure.

Module Function(s)

a) Query operator and open 3 disk images (IMGOP) the first image is the output and the second and third are input.

b) Query the operator for

- SHFT (default value of 0)
- MLTPLR (default value of 128)
- PWR (default value of 1).

c) Give the operator the choice of:

- Approving all input
- Disapproving all input
- Aborting

in the case of aborting set R2 = 2.

Error Handling

If an error occurs in opening the image disk file set RC = 2 and return to the calling module. Close all open files and delete the output file if it exists.

Calling Sequence

Call RATIN (LUN, SHFT, MLTPLR, PWR, RC).

3.3.71 COPH (1.2.4.1.5) Create Header and Log Entry

Input Name(s) and Specification

NOIMG: Number of Parent Images (Integer)

DKLUN(7): Disk Images Logical Unit numbers (Integer) 1st is the Output File the rest are for the Parent Images

OPCOD: Code Representing the operation (Integer)

1 = Ratioing of two images
2 = Classification of images
3 = Refresh to Disk
4 = Algebraic combination of images

OPDES(90): Description of the operation performed on parents (see 3.2.2) words 165-256)

Output Name(s) and Specification

ERCOD: Error Code

1 = Operation was Successful
2 = Operation was Unsuccessful

Module Function(s)

a) Allocate internal disk image header and log buffers.

b) Call SEQNO to obtain next available log sequence number (NXTSEQ) and next available log record number (NXTREC).

c) Read the date and time from the system. Store NXTSEQ, date, time fields in both the header and log buffers. Call EDIT if operator wishes to insert/modify user comment. See Tables 4 and 5 for layout of header and log file records.

d) Write the header buffer to DKLUNG (1), Record 1, the output file's header record.

e) Store the following in the log buffer: the sequence number of each parent image, NOIMG, OPCOD, OPDES.

f) Open IMAGER.LO, write the log buffer to IMAGER.LO, Record NXTREC, and close IMAGER.LO.

Error Handling

If a disk error occurs during execution inform the operator, close any file that is open and return to the caller with ERCOD = 2.

Calling Sequence

Call COPH (NOIMG, DKLUN, OPCOD, OPDES, ERCOD).

3.3.72 RDIV (1.2.4.1.3) Calculate Ratio of Two Lines of Imagery

Input Name(s) and Specification

PIX1: 1 line of NUM imagery. (Integer)

PIX2: 1 line of DENUM imagery. (Integer)

N: Number of pixels on 1 line of imagery. (Integer)

SHFT: See 3.3.70

MLTPLR: See 3.3.70

PWR: See 3.3.70

Output Name(s) and Specification

RATIO: 1 line of ratio image. (Integer)

Module Function(s)

Performs the following calculation for each line:

```
INC = .00000001
RAT(I) = (PIX1(I) + INC)/(PIX2(I) + INC)**PWR
RATIO(I) = MLTPLR * RAT(I) + SHFT + 0.5
```

Round any value less than 0 to 0 and any value greater than 255 to 255.

Error Handling

None.

Calling Sequence

```
Call RDIV (PIX1, PIX2, RATIO, N, SHFT, MLTPLR, PWR).
```

3.3.73 INCOR (1.2.4.2) Intensity Correlation

Input Name(s) and Specification

None.

Output Name(s) and Specification

None.

Module Function(s)

a) Get operator input defining 2 disk image files and any previous correlation plot file the operator wants to use. (CORIN).

b) Provide the operator with the following menu:

- Edit designated areas (EDMEN)
- List designated areas (COLST)
- Produce correlation plot (INPLT)
- Output plot file (COROP)
- Allow colour selection (PMENU)
- Return.

Error Handling

None.

Calling Sequence

Call INCOR.

3.3.74 CORIN (1.2.4.2.1) Get Correlator Input

Input Name(s) and Specification

None.

Output Name(s) and Specification

CINFO (See 3.2.2).

Module Function(s)

a) Query for the X axes image file no. (IMGOP).

b) Load that image in Refresh buffer 0 (DKREF).

c) Query for the Y axes image file no. (IMGOP).

d) Load that image in Refresh buffer 1 (DKREF).

e) Query operator for the correlation (.CO) file name and clear CINFO if file is not to be used.

f) Copy the correlation (.CO) file into CINFO if file name was specified.

g) Close all open files.

Error Handling

If a disk error occurs during the opening of a file inform the operator of the type of error and requery the operator with an option to exit. On any other disk error report it on the line printer, close all open files and return to the caller.

Callng Sequence

Call CORIN (CINFO).

3.3.75 EDMEN (1.2.4.2.2) Edit Menu

Input Name(s) and Specification

CINFO (See 3.2.2).

Output Name(s) and Specification

CINFO (See 3.2.2).

Module Function(s)

a) Show bank 0 on the monitor.

b) Display on the overlay in colour each of the previously defined regions in CINFO (DISAR).

c) Present an Edit menu to the operator with the following options:

- Toggle Input Device
- Insert Red Regions (IREGI)
- Insert Green Regions (IREGI)
- Insert Blue Regions (IREGI)
- *Delete Regions (DREGI)*
- Return.

d) Display the current device type with the menu.

Error Handling

None.

Calling Sequence

Call EDMEN (CINFO).

2 Known Limitations Exist:

a) When deleting an area entirely contained inside another area, it will be impossible to delete the inside one without also deleting the outside one.

b) When deleting overlapping areas of the same colour part of the rectangles delimiting these areas will be erased on the overlay memory. To get around this problem, exit EDMEN and re-enter.

3.3.89 CLASS (1.2.4.3) Image Classification

Input Name(s) and Specification

None.

Output Name(s) and Specification

None.

Module Function(s)

a) Query the operator for up to 6 input file images signature vectors and error vector, a confidence level and an output file image number (CLSIN).

b) Generate header information and copy it to the header and log files (COPH).

c) Calculate the threshold distance using the confidence level (THRES).

d) Form a classified image (CLIMG).

Error Handling

If a disk error occurs during the execution of CLASS control will be passed to the calling routine.

Calling Sequence

Call CLASS.

3.3.88 PMENU (1.2.4.2.6) Correlation Plot Menu

Input Name(s) and Specification

None.

Output Name(s) and Specification

None.

Module Function(s)

a) Select bank no. 3 for the pseudo colour memory and enable pseudo colour memory.

b) Give the operator a colour menu being:

Red
Green
Blue
Background
Return

The operator may select any combination of the above.

c) Write the pseudo colour memory look-up tables according to the operator selection. The intensity of the colour will be 128 and the intensity of the background 64, the intensity for the calibrated axes will be set to 192.

d) Repeat b) and c) until return is selected in b).

Error Handling

None.

Calling Sequence

Call PMENU.

3.3.87 COROP (1.2.4.2.5) Perform Correlator Output

Input Name(s) and Specification

CINFO (see 3.2.2).

Output Name(s) and Specification

None.

Module Function(s)

a) Ask the operator for an output filename, append the extension ".CO" to the filename.

b) If the file already exists, ask operator permission to delete it. On a negative answer reprompt the operator. On a positive answer delete the old file and continue.

c) Write the plot info (CINFO) to the disk file.

Error Handling

If an error occurs with a disk file report that error on the line printer and transfer control to the calling program.

Calling Sequence

Call COROP (CINFO).

Know Limitation:

No returns are possible without creating a disk file.

3.3.86 COROL (1.2.4.2.4.3) Produce Correlation Plot

Input Name(s) and Specification

AREA(4): Limit are a position in order XMIN, XMAX, YMIN, YMAX.

Output Name(s) and Specification

None.

Module Function(s)

a) Clear refresh Buffer no. 3.

b) Form calibrated axes of 256 x 256 and identify them with "BUFFER 0" for the X axis and "BUFFER 1" for the Y axis.

c) Transpose the coordinate system of AREA into the refresh buffer coordinate system.

d) For each pixel pair inside the limited area in the two input images, set the intersection point of the Intensity vs. Intensity in refresh buffer no. 3 to an Intensity equal.

1 = Red
2 = Green
3 = Blue
4 = Background

This code output will be determined be the value of the corresponding pixel in refresh buffer no. 2 as defined in DESIG (3.3.84).

Error Handling

None.

Calling Sequence

Call COROL (AREA).

3.3.85 LIMIT (1.2.4.2.4.2) Select Limited Region

Input Name(s) and Specification

DVUSE: Input device to be used

1 = Keyboard
2 = Joystick
3 = User selection.

Output Name(s) and Specification

AREA(4): Limit area position in order XMIN, XMAX, YMIN, YMAX.

Module Function(s)

a) If user selection is requested query the operator to find if either a limited area or the full image is requested.

b) If the entire image is requested set AREA (1) = 0, AREA (2) = 639, AREA (3) = 0, AREA (4) = 379 and exit.

c) Else turn the overlay on.

d) If user selection is requested query the operator for the device selection Keyboard or Joystick.

e) Read the limits of the area (KEYXY or JOYXY) if the coordinates are outside the image, get a new location from the selected device.

f) Show the selected image perimeter in white (SHWRG).

g) Get operator's approval for the selected region.

h) If the operator refused, erase the region from the monitor (SHWRG) and read another region (step e).

Error Handling

None.

Calling Sequence

Call LIMIT (AREA, DVUSE).

3.3.84 DESIG (1.2.4.2.4.1) Form Designated Area

Input Name(s) and Specification

CINFO (See 3.2.2).

Output Name(s) and Specification

None.

Module Function(s)

This module will first clear refresh no. 2 and then, examine CINFO and for each defined area fill the corresponding pixels of refresh buffer no. 2 with a code defining the colour as follows:

1 = Red
2 = Green
3 = Blue

If a pixel is already defined by a code, this code is to be preserved and the new colour added to it.

Error Handling

None.

Calling Sequence

Call DESIG (CINFO).

3.3.83 INPLT (1.2.4.2.4) Perform Correlation Plot

Input Name(s) and Specification

CINFO (see 3.2.2).

Output Name(s) and Specification

None.

Module Function(s)

a) Construct an image in refresh buffer no. 2 where each pixel value indicates an area type. (DESIG).

b) Enable bank 0.

c) Get the limit for the area to be plotted (LIMIT).

d) Erase the rectangle delimiting the area (SHWMG).

e) Produce a correlation plot in refresh buffer no. 3 (COROL).

Error Handling

None.

Calling Sequence

Call INPLT (CINFO).

3.3.82 COLST (1.2.4.2.3) List Designated Area

Input Name(s) and Specification

CINFO (See 3.3.2).

Output Name(s) and Specification

None.

Module Function(s)

This module will present to the operator the X & Y coordinates and the colour of all defined designated areas.

Error Handling

None.

Calling Sequence

Call COLST (CINFO).

3.3.81 DREGI (1.2.4.2.2.3) Delete an Area

Input Name(s) and Specification

CINFO: (See 3.2.2)

DVCOD: Device used for input

1 Joystick
-1 Keyboard

Output Name(s) and Speciiication

CINFO (See 3.3.2).

Module Function(s)

a) Get X & Y position (JOYXY, KEYXY). If the cursor was positioned outside the image return to the calling module.

b) Examine CINFO to find if the X & Y position is inside a designated area, if no designated areas are found repeat a).

c) For the designated areas found, delete the square in the overlay memory (SHWRG). Output on console the deleted rectangle coordinates.

d) Repeat a), b) and c).

Error Handling

None.

Calling Sequence

Call DREGI (CINFO, DVCOD).

3.3.80 JOYXY (1.2.4.2.2.1.1) Determine X,Y Position of Joystick

Input Name(s) and Specification

None.

Output Name(s) and Specification

XPOS: X position of the cursor at interrupt time in image coordinates (Integer).

YPOS: Y position of the cursor at interrupt time in image coordinates (Integer).

EXCOD: Exit code

1 = Cursor was inside the image.
2 = Cursor was outside the image.

Module Function(s)

a) Wait for a cursor interrupt, and read its position.

b) Translate the cursor position into image coordinates.

c) Set proper exit code.

Error Handling

None.

Calling Sequence

Call JOYXY (XPOS, YPOS, EXCOD).

3.3.79 KEYXY (1.2.4.2.2.1.3) Get X,Y Location from Keyboard

Input Name(s) and Specification

None.

Output Name(s) and Specification

Same as 3.3.80.

Module Function(s)

Same as 3.3.80 except input of X & Y are from the keyboard.

Error Handling

None.

Calling Sequence

Call KEYXY (XPOS, YPOS, EXCOD).

3.3.78 IREGI (1.2.4.2.2.2.) Insert an Area

Input Name(s) and Specification

CINFO: (See 3.2.2)

COCOD: Colour code

1 Red
2 Green
3 Blue.

DVCOD: Device used for input

1 Joystick
-1 Keyboard

Output Name(s) and Specification

CINFO (See 3.2.2).

Module Function(s)

a) Get two X & Y positions (JOYXY or KEYXY). If the cursor was positioned outside the image, return to the calling module.

b) For the newly defined area, draw a rectangle of the appropriate colour on the overlay memory (SHWRG).

c) Find a blank space in CINFO and put in the newly defined area.

d) Repeat a, b, and c.

Error Handling

Report to the operator if the 10 designated areas are filled for that colour and return to the calling module.

Calling Sequence

Call IREGI (CINFO, COCOD, DVCOD).

3.3.77 SHWRG (1.2.4.2.2.1.2) Show Designated Area

Input Name(s) and Specification

ARPOS: Area position coordinates (Integer) in order XMIN, XMAX, YMIN, YMAX.

DECOD: Function code

1 = Draw in red
2 = Draw in green
3 = Draw in blue
4 = Draw in white
5 = Draw in yellow
6 = Draw in cyan
7 = Draw in magenta
-1 = Erase in red
-2 = Erase in green
-3 = Erase in blue
-4 = Erase in white
-5 = Erase in yellow
-6 = Erase in cyan
-7 = Erase in magenta.

Output Name(s) and Speciiication

None.

Module Function(s)

a) Enable overlay display.

b) Select the overlay to display bits 0,1,2 respectively on the red, green and blue guns.

c) Write the rectangle as defined by ARPOS coordinates to the appropriate bits of the overlay memory to produce the colour defined by DECOD.

Error Handling

None.

Calling Sequence

Call SHWRG (ARPOS, DECOD).

3.3.76 DISAR (1.2.4.2.2.1) Display All Regions

Input Name(s) and Specification

CINFO (See 3.2.2).

Output Name(s) and Specification

None.

Module Function(s)

a) Clear overlay memory and select overlay on.

b) Examine CINFO and find all defined (non zero) position(s).

c) For all defined position(s) draw a rectangle in the appropriate colour in the overlay memory (SHWRG).

Error Handling

None.

Calling Sequence

Call DISAR (CINFO).

3.3.90 CLSIN (1.2.4.3.1) Input Classification Parameters

Input Name(s) and Specification

None.

Output Name(s) and Specification

NIMG: Number of image file(s).

SIGVE(6): Target signature (real).

ERRVE(6): Error vector (real).

INLUN(6): Input disk file logical unit(s) no. (Integer).

OVLUN(6): Output disk file logical unit no. (Integer).

CONLV: Confidence level (real).

ERCOD: Error code (Integer)

1 = Successful
2 = Unsuccessful.

Module Function(s)

a) Get from the operator:

- NIMG acceptable range 1 to 6
- SIGVE (1 to NIMG) acceptable range 0 to 255
- ERRVE (1 to NIMG) default value of 1 each
- CONLV acceptable range 0 > CONLV => 0.9999.

b) Open NIMG file(s) for input (IMGOP).

c) Open 1 image file for output (IMGOP).

d) Get operator approval on all input. If approval is denied by the operator close all open files and go back to a). If the operator choses to exit set ERCOD = 2 and close all open files and return to the calling module.

Error Handling

This module will check every operator input to be in range. If the input is out of range it will output a console error message and repeat the input question. If a disk error occurs it will print an error message on the line printer, close all open files and return to the caller with ERCOD = 2.

Calling Sequence

Call CLSIN (NIMG, SIGVE, ERRVE, INLUM, OVLUN, CONLV, ERCOD).

3.3.91 THRES (1.2.4.3.2) Calculate Threshold Distance

Input Name(s) and Specification

NIMG: Number of image file(s).

CONLV: Confidence level (real)

Output Name(s) and Specification

TSQ: Threshold distance squared (real).

Module Function(s)

This module will calculate TSQ using the method described in applicable document no. 2.

Error Handling

None.

Calling Sequence

Call THRES (NIMG, CONLV, TSQ).

3.3.92 CLIMG (1.2.4.3.3) Form Classified Image

Input Name(s) and Specification

NIMG: Number of image file(s).

SIGVE(6): Signature vector (real).

ERRVE(6): Error vector (real).

INLUN(6): Input disk file(s) logical unit no(s).

OVLUN: Output disk file logical unit no.

TSQ: Distance T**2 (real).

Output Name(s) and Specification

ERCOD: Error code

1 = Successful
2 = Unsuccessful.

Module Function(s)

This module will form a classified image in the following way:

a) Calculate D**2 for 1 line (DSQLN).

b) Calssify 1 line (CLSLN).

c) Output the line of classified data to the output image disk file.

d) Repeat a, b and c for all the lines.

e) Close all open files.

Error Handling

On an disk error report the error on the line printer, close all open files and return to the calling routine with ERCOD = 2.

Calling Sequence

Call CLIMG (NIMG, SIGVE, ERRVE, INLUN, OVLUN, TSQ, ERCOD).

3.3.93 DSQLN (1.2.4.3.3.1) Calculate Square of Distance for One Line

Input Name(s) and Specification

NIMG: Number of image file(s).

INLUN(6): Input file logical disk unit no.

SIGVE(6): Signature vector (real).

ERRVE(6): Error vector (real).

Output Name(s) and Specification

LNDSQ(640): 1 line of distance square data (real).

ERCOD: Error code

1 = Successful
2 = Disk error.

Module Function(s)

a) Clear LNDSQ.

b) Input one line of Pixel data.

c) Calculate the square of the distance for each pixel on the line using the formula in applicable document no. 2.

d) Repeat b) and c) NIMG times.

Error Handling

If any disk error occurs during DSQLN, report the error on the line printer, set ERCOD = 2 and return to the calling module.

Calling Sequence

Call DSQLN (NIMG, INLUN, SIGVE, ERRVE, LNDSQ, ERCOD).

3.3.94 CLSLN (1.2.4.3.3.2) Classify One Line

Input Name(s) and Specification

LNDSQ(640): 1 line of distance square data (real).

TSQ: Threshold distance square (real).

Output Name(s) and Specification

LNCLD(640): 1 line of classified imagery (integer).

Module Function(s)

This module will set the intensity to 128 in LNCLD if the corresponding pixel in LNDSQ is less or equal to TSQ and to zero otherwise.

Error Handling

None.

Calling Sequence

Call CLSLN (LNDSQ, TSQ, LNCLD).

3.3.95 COLCO (1.2.4.4) Colour Composite Image

Input Name(s) and Specification

None.

Output Name(s) and Specification

None.

Module Function(s)

a) Clear Refresh Buffer 1, 2 and 3.

b) Connect the Red gun to RB1
Connect the Green gun to RB2
Connect the Blue gun to RB3

c) Get a disk file number for each colour/Refresh Buffer (IMGOP).

d) Load each Image into the appropriate Refresh Buffer (DKREF).

Error Handling

If IMGOP returns an unsuccessful error code skip over that colour. If an error occurs in d) return to the calling module.

Calling Sequence

Call COLCO.

3.3.96 ALGCOMB (1.2.4.5) Algebraic Combinations of Images

Input Name(s) and Specification

None.

Output Name(s) and Specification

None.

Module Function(s)

a) Input an equation (EQINPUT) (ASCII characters).

b) Transform the equation to reverse polish notation (RPOLISH).

c) Transform the reverse polish notation to an assembly language subroutine and form the real and integer constant stacks (COMPILE).

d) Open input image files (LUN 1-6) and Skip header records.

e) Open temporary output file ($ALGC) LUN = 14.

f) Ask the operator for the image buffer bank number for display.

g) If bank no. in the range 0-3 then set up for DISPLAY.

h) Read next line of image data into image buffer for each resulting image.

i) Process the assembly program (EQPROC) for 1 line.

j) Write the resulting image line to the temporary output file.

k) Write the resulting image line to the image buffer of bank no. positive.

l) Repeat h) to k) for 480 lines.

m) Ask operator if he wants to save the resulting image.

n) If he wants the result saved then

   i) Create output image file header log file entry (COPH).

ii) Write refresh buffer file header (DIMXX where XX = 17 + bank no.) (ODKWT).

iii) Rename Temp. Output file to requested Image File (GETNM).

Error Handling

Return to caller if any error is reported by any module called in ALGCOMB.

Calling Sequence

Call ALGCOMB.

3.3.97 EQINPUT (1.2.4.5.1) Equation Input Package

Input Name(s) and Specification

None.

Output Name(s) and Specification

EQBUF(80): Equation Buffer (ASCII) terminated with a null character.

ERCOD: Error Code

1 = Successful
2 = Unsucessful

Module Function(s)

a) Query the user for the equation to be used.

b) Get operator approval on the equation offer. Yes, No, or Exit choice.

c) Check the input equation for syntax.

Error Handling

Check the equation inputed for the following list of errors:

1 - Only acceptable characters are in the input equation.

Acceptable characters are:

+ plus sign
- minus sign
/ divide sign
< angle bracket for image disk file No.ID
> angle bracket for image disk file No.ID
* multiplication
0-9 numeric
. decimal point
= equal sign
(round bracket
) round bracket
Space ignored

2 - The validity of <xx> where xx is one or two digit(s) representing a disk image file number and in the range 1 to 16.

3 - That the equation starts with <xx> =.

4 - That the same amount of "(" and ")" are present.

5 - That only one = sign is present.

6 - That 2 "."'s are separated by at least one +, -, /, * sign.

7 - That an (is always preceded by a +, -, /, * or =.

8 - That a ")" is always followed by a +, -, / or * sign or terminator.

9 - That an (is not followed by a / or * sign.

10 - That a) is not preceded by a +, -, / or * sign.

11 - That, <xx> is followed or preceded by a +, -, /, =, *, (or) sign.

12 - That there is no more than 2 "*"'s together.

13 - That the following pairs of operators are not in sequence:

(+,*), (-,*), (1,*), (+,1), (-,1), (*,1).

Note: Redundant "+"'s will be removed and the negative operator '-N' will be replaced with '(0 - N)'.

Any error will be displayed on the console and the operator given another chance to enter the equation. If the exit route is taken in b) set ERCOD to 2, otherwise set it to 1.

Calling Sequence

```
Call EQINPUT (EQBUF, ERCOD).
```

3.3.98 RPOLISH (1.2.4.5.2) Reverse Polish Notation Transformation

Input Name(s) and Specification

EQBUF(80): Equation Buffer (ASCII) terminated by a null character.

Output Name(s) and Specification

POBUF(100): Polish notation buffer (IMAGER Internal Code).

ERCOD: Error code

1 = Successful
2 = Unsuccessful

Module Function(s)

Express the equation in the reverse polish notation mode using the following code to express the result:

- precede any entry in POBUF by a code representing the type of entry that will be following

  24. - a real number (4 bytes)
  25. - an integer number (2 bytes)
  26. - a disk image file number (1 byte)
  03. - a multiplication operation
  08. - a division operation
  04. - an addition operation
  06. - a subtraction operation
  19. - an exponential operation
  22. - an equal sign

- all these codes are stored in byte format to minimize storage room.

Error Handling

Any error in transforming the equation to reverse polish notation will be flagged, the process stopped and a 2 error code returned.

Calling Sequence

Call RPOLISH (EQBUF,POBUF,ERCOD).

3.3.99 COMPILE (1.2.4.5.3) Form Assembly Language Program

Input Name(s) and Specification

POBUF(100): Polish notation buffer (IMAGER Internal Code).

Output Name(s) and Specification

NOIMG: No. of parent image files to be processed (Integer).

PROGA(1000): Program buffer (Assembler Code).

REALC(100): Real constant buffer (Real).

INTGC(100): Integer constant buffer (Integer).

DKNO(7): Disk image file no.'s (Integer).

ERCOD: Error code

1 = Successful
2 = Unsuccessful

Module Function(s)

a) Replace all disk image file numbers in POBUF by a logical unit number. Where LUN = 14 for output and LUN = 1 to 6 for input. Store the disk image numbers in DKNO such that DKNO(2 → 7) will be for input LUN 1 → 6 respectively. If an image file number appears twice it will receive the same LUN.

b) Replace all constants in POBUF by the index of their storage position in REALC or INTGC.

c) Store in PROGA the assembly code to

- Perform the operation defined in POBUF for 1 line of image data using either integer mode when operating on integer only and floating point when at least one operand is real. Get all constants from their proper stacks and disk image data from the appropriate displacement in the image buffer. Store all intermediate results on the stack.

  The result of the operation will be stored in the 7th row of image buffer and should be no larger than 255 and not less than 0.

- Return

Error Handling

Flag any buffers that overflow and return to the calling module. Set ERCOD = 2.

Calling Sequence

Call COMPILE (POBUF, NOIMG, PROGA, REALC, INTGC, DKNO, ERCOD).

3.3.110 LSLICE (1.2.5.3.2) Load Pseudo-Colour Table for Slice Colouration

Input Name(s) and Specification

NOSLC: Number of slices max of 6 (Integer)

SLIVE(5): Slice level vector (Integer)

Output Name(s) and Specification

None.

Module Function(s)

For every slice selected load the matching section of the three Pseudo-colour Tables using the data supplied in the following table:

NOSLC = 2 3 4 5 6	Slice Colours	RED TABLE	GREEN TABLE	BLUE TABLE
1 1 1 1 1	Black	0	0	0
2 2 2	Blue	0	0	255
Interval Number 3	Green	0	255	0
3 4	Yellow	255	255	0
2 3 4 5	Red	255	0	0
2 3 4 5 6	White	255	255	255

Error Handling

None.

Calling Sequence

Call LSLICE (NOSLC, SLIVE).

3.3.109 SSLICE (1.2.5.3.1) Select Slice Level

Input Name(s) and Specification

HISVE(0:255): Histogram vector (Real)

Output Name(s) and Specification

NOSLC: Number of slices (Integer)
SLIVE(5): Slice level vector (Integer)

Module Function(s)

a) Generate the histogram (HPLOT (HISVE,-1)).

b) Ask the operator which input device to be used, keyboard or joystick. If the keyboard is selected query the operator for the slice level directly. If the joystick is selected query the operator using JOYXY module and use only the X value obtained.

c) Get all operator slice selection (SLIVE), the last slice ends at 255.

d) Replot the histogram in the appropriate colours using the methods described in GRAPH.

NOSLC = 2 3 4 5 6	Data	Colour
1 1 1 1 1	0	black
2 2 2	4	blue
3	2	green
3 4	3	yellow
2 3 4 5	1	red
2 3 4 5 6	7	white

Interval Number

Error Handling

Disregard any slice selection that is less or equal to the previous slice selected also disregard any joystick selection that is not situated inside the histogram displayed.

Calling Sequence

Call SSLICE (NOSLC, HISVE, SLIVE).

3.3.108 SLICE (1.2.5.3) Level Slice Colouration Package

Input Name(s) and Specification

HISVE(0:255): Histogram vector (Real)

Output Name(s) and Specification

None.

Module Function(s)

a) Query the operator for the number of slices to be used, acceptable range 2 to 6.

b) Select the slice level (SSLICE).

c) Load the pseudo-colour memory according to the slice levels chosen in b) (LSLICE).

d) Offer the following menu:

- Display Image with Colour Slice (DS)
- Display Original Image (DO)
- Return to SSA menu (RT)

The original image is stored in refresh buffer 0 and the colour sliced image is shown through the three Pseudo-Colour Tables.

This module will reject any operator input that is not inside the acceptable range, or not one of the menu's legal codes.

Calling Sequence

Call SLICE (HISVE).

j) Form the histogram with intensity equal to 7. Compute the height of the line to be drawn by multiplying the proper VECTO element by SCALFACT.

Error Handling

None.

Calling Sequence

Call GRAPH (VECTO, XSLOC, YSLOC).

3.3.107 GRAPH (1.2.5.3.1) Histogram Plotting Routine

Input Name(s) and Specification

VECTO(0:255): Vector to be plotted (Real)

XSLOC: X axis starting position (Integer)

YSLOC: Y axis starting position (Integer)

Output Name(s) and Specification

None.

Module Function(s)

a) Find the highest element of VECTO.

b) Calculate SCALFACT (200/Highest element in VECTO).

c) Draw the axes with intensity equal to 3 (256 in x and 200 in y).

d) Mark the X axis with a vertical line of 6 (Cyan) pixels long, these marks will be spaced by 16 pixels the first one being 0 with intensity equal to 3 (Yellow).

e) Mark the Y axis with a horizontal line of 6 pixels long, these marks will be spaced by 20 pixels and the first one is at 0 with intensity equal to 3.

f) Identify the X axis Scale by writing the appropriate pixel intensity under every second mark, place the first one at zero with the data equal to 1 (yellow).

g) Form an array containing the ASCII data for the Y axis. Compute the values to be used for every coordinate position by dividing the actual position by SCALFACT. These values should be expressed in "F" format.

h) Use that array to identify the Y axis with data equal to 1.

i) Identify the X axis with INTENSITY and the Y axes with POPULATION, use a character size of 2 for that step and set the data equal to 1.

3.3.106 HPLOT (1.2.5.2.3) Plot Histograms

Input Name(s) and Specification

HISVE(0:255): Histogram vector (Real)

EQHIS(0:255): Equalized histogram vector (Real)

Output Name(s) and Specification

None.

Module Function(s)

a) Fill the overlay memory with intensity 6

b) Connect the Red Gun to the overlay bit No. 0
Connect the Green Gun to the overlay bit No. 1
Connect the Blue Gun to the overlay bit No. 2

c) Fill bank 3 calibrate table with all zeros (VTABLE).

d) Display bank 3 through its video lookup table and turn the overlay ON.

e) If the first element of EQHIS is equal to -1 plot only the histogram vector in the center of the display (GRAPH).

f) If the first element of EQHIS is not -1 plot both histogram one underneath the other (GRAPH).

Error Handling

None.

Calling Sequence

Call HPLOT (HISVE, EQHIS).

3.3.105 EQUIL (1.2.5.2.2) Calculate Equalized Histogram

Input Name(s) and Specification

HISVE(0:255): Histogram vector (Real)

TRSVE(0:255): Transformation vector (Integer)

Output Name(s) and Specification

EQHIS(0:255): Equalized histogram vector (Real)

Module Function(s)

This module will calculate an equalized histogram Vector using the following method:

EQHIS (I) = HISVE(J) + EQHIS (I)

where I = 0,255 and for all "J" (0,255) such that

TRSVE (J) = I

Prior to computing the equalized histogram, EQHIS should be initialized so that all components are zero.

Error Handling

None.

Calling Sequence

Call EQUIL (HISVE, TRSVE, EQHIS).

Calling Sequence

```
Call TRANS (AREA, HISVE, TRSVE).
```

3.3.104 TRANS (1.2.5.2.1) Calculate Transformation Vector

Input Name(s) and Specification

AREA(4): Limit area position in order XMIN, XMAX, YMIN, YMAX.

Output Name(s) and Specification

HISVE(0:255): Histogram vector; population for each intensity value indexed by intensity (Real).

TRSVE(0:255): Transformation vector; new intensity to be used to equalize the original image, indexed by the intensity to be replaced (Integer).

Module Function(s)

a) Calculate the Histogram vector (HISVE) by counting the number of times each pixel intensity is present inside the defined area in refresh buffer 0.

b) Calculate the un-normalized transformation vector UNNOT (0:255) (REAL). This vector will be calculated the following way:

UNNOT(I) = UNNOT(I-1) + HISVE(I)

where I = 0,255

care should be exercised in the case of I = 0.

c) Normalize the transformation vector by doing the following computation:

TRSVE(I) = (UNNOT(I)/UNNOT(255)) x 255

where I = 0,255

Only the integer part should be retained in TRSVE

d) Load the transformation vector (TRSVE) into the calibrate table of refresh buffer 0. (Call TABLE).

Error Handling

None.

Calling Sequence

Call HISTO (HISVE).

3.3.103 HISTO (1.2.5.2) Histogram Equalization Package

Input Name(s) and Specification

None.

Output Name(s) and Specification

HISVE(0:255): Histogram Vector (Real)

Module Function(s)

a) Get from the operator the coordinate of the area to be processed, this entry should be specfied either by keyboard or by joystick (LIMIT).

b) Calculate the histogram vector and the transformation vector (TRANS).

c) Calculate the equalized histogram vector, (EQUIL).

d) Offer the following menu:

- Display Original Image (DO)
- Display Equalized Image (DE)
- Display Histograms (DH)
- Return to SSA menu (RT)

For each of the displayed functions enable the proper memory bank to be displayed.

The Original Image is stored in bank 0.

The Equalized Image is stored in bank 0 through the calibrate table.

The histograms are stored in the overlay memory and should be displayed over bank 3 going through a nulled calibrate table.

Error Handling

This module will ignore any operator input that is not one of the menu's legal codes.

3.3.102 RBOLD (1.2.5.1) Load Refresh Buffer 0

Input Name(s) and Specification

None.

Output Name(s) and Specification

None.

Module Function(s)

a) Ask the user for an image disk file number and open that file (IMGOP).

b) Transfer the opened disk file to refresh buffer 0 (DKREF).

Error Handling

If any disk error occurs during the execution of this module, control will be passed to the calling module.

Calling Sequence

Call RBOLD.

3.3.101 SSA (1.2.5) Mono Spectral Analysis Menu

Input Name(s) and Specification

None.

Output Name(s) and Specification

None.

Module Function(s)

a) Clear HISVE array (length 256 words)

b) Present the following menu

- Load Image (LI)
- Histogram Equalization (HE)
- Level Slice (SL)
- Zooming (ZO)
- Pixel Intensity Dump (DD)
- Return to Main Menu (RT)

and call the appropriate subroutine depending upon the user selection.

c) If 'LI' is selected, call RBOLD
If 'HE' is selected, call HISTO
If 'SL' is selected, call SLICE
If 'ZO' is selected, call ZOOM
If 'DD' is selected, call DDUMP

Error Handling

This module will ignore any illegal operator input.

Calling Sequence

Call SSA.

3.3.100 EQPROC (1.2.4.5.4) Process Equation Routine (Driver for PROGA)

Input Name(s) and Specification

PROGA(1000): Program buffer (Assembly Code)

REALC(100): Buffer of real constants (Real)

INTGC(100): Buffer of integer constants (Integer)

IMBUF(2240): Image file buffer (640 x 7 bytes)

Output Name(s) and Specification

None.

Module Function(s)

a) Set up data areas etc.

b) Call the subroutine whose starting address is the same as the first address of PROGA.

c) Upon return from the above subroutine the resulting data is contained in the last line of the image buffer.

Calling Sequence

Call EQPROC (PROGA, REALC, INTGC, IMBUF).

3.3.111 ZOOM (1.2.5.4) Zoom Package

Input Name(s) and Specification

None.

Output Name(s) and Specification

None.

Module Function(s)

a) Query for the magnification factor and the central point of the area to be zoomed (ZOOMI).

b) Magnify the image in refresh buffer 0 into refresh buffer 1. (MAGNI).

c) Print selected pixel coordinates on the terminal (PIXCO).

Error Handling

None.

Calling Sequence

Call ZOOM.

3.3.112 ZOOMI (1.2.5.4.1) Zoom Image Input Package

Input Name(s) and Specification

None.

Output Name(s) and Specification

XLOC: X Location of the central point in image coordinate (Integer).

YLOC: Y Location of the central point in image coordinate (Integer).

MAFAC: Magnification factor (Integer).

Module Function(s)

a) Show bank.

b) Query the operator for the magnification factor (MAFAC) acceptable range 2 to 10.

c) Prompt operator for the input device (joystick or keyboard) to be used for defining the zoomed image central point.

d) Get the central points of zoom (JOYXY or KEYXY). If chosen point is outside the image repeat c) until valid point is chosen by the operator.

e) Give the operator the opportunity to accept or reject the current selection.

Error Handling

Repeat every query whose reply is not in range.

Calling Sequence

Call ZOOMI (XLOC, YLOC, MAFAC).

3.3.113 MAGNI (1.2.5.4.2) Image Magnification

Input Name(s) and Specification

XLOC: X location of the central point of zoom (Integer)

YLOC: Y location of the central point of zoom (Integer)

MAFAC: Magnification Factor (Integer)

Output Name(s) and Specification

MINX: X coordinate of left side of image to be magnified. (Integer).

MINY: Y coordinate of top of image to be magnified. (Integer).

Module Function(s)

a) Clear the disk header file for refresh buffer 1.

b) Display refresh buffer 1 and turn overlay off.

c) Calculate selected area using XLOC, YLOC, MAGNI.

d) Read data from refresh buffer 0 and MAGNIFY it and store the result into refresh buffer 1 using the following function:

Write pixel intensities of RBO (I,J) into RBI (X,Y)

where: (I,J) represents a point in the selected area in c)

and $I * MAFAC < X < (I + 1) * MAFAC$

$J * MAFAC < Y < (J + 1) * MAFAC$

where: (I,J) references the coordinates of the selected area with the point (0,0) in the upper left hand corner of the selected area.

Error Handling

Set Intensities from negative coordinate regions to 0.

Calling Sequence

Call MAGNI (XLOC, YLOC, MAFAC, MINX, MINY).

3.3.114 PIXCO (1.2.5.4.5) Identify Pixel Coordinates

Input Name(s) and Specification

MINX: X coordinate of left side of magnified image (Integer).

MINY: Y coordinate of top side of magnified image (Integer).

MAFAC: Magnification Factor (Integer).

Output Name(s) and Specification

None.

Module Function(s)

a) Erase bit 3 of the overlay memory.

b) Connect the Red gun on bit number 3 of the Overlay and turn the Green and Blue gun off.

c) Prompt operator to use joystick to center point of interest. Get the coordinate location (JOYXY).

d) Write vectors forming a 7 x 7 red cross over the point selected in c).

e) In order to be able to correlate the cross with the printed data identify the cross by writing a number representing the iteration number in the overlay memory in red.

f) Calculate the un-zoomed image pixel coordinates and write them on the console along with the iteration and the pixel intensity read from Refresh Buffer 0.

g) Repeat steps c) to f) until an out of image coordinate is selected.

Error Handling

None.

Calling Sequence

Call PIXCO (MINX, MINY, MAFAC).

3.3.115 DDUMP (1.2.5.5) Pixel Intensity Dump Package

Input Name(s) and Specification

None.

Output Name(s) and Specification

None.

Module Function(s)

Set image limits (XMIN, XMAX, YMIN, YMAX) to full image.

Clear the overlay. Display the following menu:

- Select a New Refresh Buffer (SB)
- Toggle Input Device (TD)
- Select a Region (SR)
- Calculate Mean and Standard Deviation (ST)
- Print Pixel Data (PD)
- Print Header Data (PH)
- Return to SSA menu (RT)

Display the following information beneath the menu: the refresh buffer in use, the input device in use and the (XMIN, XMAX, YMIN, YMAX) coordinates defining the currently selected region.

The default refresh buffer is 0. The default input device is the joystick.

Call LIMIT to allow the operator to select a region.
Call STATS to calculate and print the mean and standard deviation.
Call PDATA to print the pixel data.
Call PHEAD to print the header data.

Error Handling

This module will reject any operator input that is not one of the menu's legal code.

Calling Sequence

Call DDUMP.

3.3.116 STATS (1.2.5.5.1) Calculate Mean and Standard Deviation

Input Name(s) and Specification

RBUSE: Refresh buffer used (Integer).

AREA(4): Area of interest in order XMIN, XMAX, YMIN, YMAX.

Output Name(s) and Specification

None.

Module Function(s)

a) Calculate the sum of all pixel intensities (SP) and the sum of the square of the pixel intensities (SPSQ) for all pixels inside AREA. Compute N, the total number of pixels inside AREA.

b) Calculate the Mean and Standard Deviation as follows:

MEAN = SP/N
Standard Deviation = SQRT(SPSQ - (MEAN*SP)/(N - 1)).

c) Print the values of Mean and Standard deviation and N on the console and on the line printer.

Error Handling

None.

Calling Sequence

Call STATS (RBUSE, AREA).

3.3.117 PDATA (1.5.5.5.2) Print Pixel Data

Input Name(s) and Specifications

RBUSE: Refresh Buffer in Use (Integer)

AREA(4): Area of Interest XMIN, XMAX, YMIN, YMAX (Integer).

Output Name(s) and Specification

None.

Module Function(s)

a) Output a header indicating the position of the data in refresh buffer coordinates.

b) Read the appropriate lines of the refresh buffer and print the line number and the pixel intensity on the line printer.

Error Handling

None.

Calling Sequence

Call PDATA (RBUSE, AREA).

3.3.118 PHEAD (1.2.5.5.3) Print Image Header

Input Name(s) and Specification

RBUSE: Refresh buffer in Use (Integer)

Output Name(s) and Specification

None.

Module Function(s)

a) Open the header file (ODKRD).

b) Print the header data (LIST) on the console and on the line printer.

Error Handling

If any error code is generated by ODKRD or LIST return to the calling module.

Calling Sequence

Call PHEAD (RBUSE).

3.3.119 MSG (1.2.4.1.1.1) Output a Message on the Terminal or on the Line Printer

Input Name(s) and Specification

MNUM: Message number (Integer)

OPCOD: Code to specify the destination of the message.

1 - to the screen
2 - to line 23 of the screen
3 - to line printer.

ARG1 to ARGn: Integer arguments to be used with the message format read from the message file. Up to 9 arguments can be specified.

Output Names(s) and Specification

None.

Module Function(s)

a) Open IMAGER.ER the message file.

b) Remember the current console cursor position and move the cursor on the last line (line 23) if OPCOD = 2.

c) Read from the message file the message associated with MNUM into an array (sec. 3.2.4). This message is used as a FORTRAN FORMAT for a WRITE command.

d) Set up to print according to the number of arguments read from IMAGER.ER.

e) Output the message on the chosen device.

f) Move the cursor back to it's original position if OPCODE = 2.

g) Close IMAGER.ER.

Error Handling

If any disk error occurs during the execution of this module inform the operator via the line printer and return to the calling module. Do not use MSG.

Calling Sequence

```
Call MSG (MNUM, OPCOD, [,ARG1[,...[,ARGn]]]).
```

UNCLASSIFIED
Security Classification

DOCUMENT CONTROL DATA - R & D
(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1 ORIGINATING ACTIVITY
DEFENCE RESEARCH ESTABLISHMENT OTTAWA
NATIONAL DEFENCE HEADQUARTERS
OTTAWA, CANADA K1A 0Z4

2a. DOCUMENT SECURITY CLASSIFICATION
UNCLASSIFIED

2b. GROUP

3 DOCUMENT TITLE
DESIGN SPECIFICATIONS FOR IMAGER:
THE DREO IMAGE PROCESSING SYSTEM

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
TECHNICAL NOTE

5. AUTHOR(S) (Last name, first name, middle initial)
BROCHU, C. and
BALL, R., LIM, K., SORENSEN, S. OF ROY BALL ASSOCIATES LTD.

6. DOCUMENT DATE
SEPTEMBER 1983

7a. TOTAL NO. OF PAGES
158

7b. NO. OF REFS

8a. PROJECT OR GRANT NO
33D

9a. ORIGINATOR'S DOCUMENT NUMBER(S)
DREO TN 82-15

8b. CONTRACT NO

9b. OTHER DOCUMENT NO.(S) (Any other numbers that may be assigned this document)

10 DISTRIBUTION STATEMENT
UNLIMITED DISTRIBUTION

11 SUPPLEMENTARY NOTES

12. SPONSORING ACTIVITY

13 ABSTRACT

This document enunciates the detailed design specifications for the development of image processing software (first three releases) for the DREO Image Processing System consisting of a Data General Eclipse S-130 computer equipped with a NORPAK RGP-3050 Image Processor. A Top-Down structured design technique approach is used. It is composed of data and file formats, structure charts and programming specifications for all modules. These modules allow image transfer and manipulation, system parameters bookkeeping and multispectral and monospectral image analysis techniques.

DSIS
77-06

UNCLASSIFIED

Security Classification

KEY WORDS

DIGITAL IMAGE PROCESSING SOFTWARE
MULTISPECTRAL INFRARED LINE SCANNER IMAGE ANALYSIS

INSTRUCTIONS

ORIGINATING ACTIVITY Enter the name and address of the organization issuing the document.

DOCUMENT SECURITY CLASSIFICATION Enter the overall security classification of the document including special warning terms whenever applicable.

GROUP Enter security reclassification group number. The three groups are defined in Appendix 'M' of the DRB Security Regulations.

DOCUMENT TITLE Enter the complete document title in all capital letters. Titles in all cases should be unclassified. If a sufficiently descriptive title cannot be selected without classification, show title classification with the usual one-capital-letter abbreviation in parentheses immediately following the title.

DESCRIPTIVE NOTES Enter the category of document, e.g. technical report, technical note or technical letter. If appropriate, enter the type of document, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.

AUTHOR(S): Enter the name(s) of author(s) as shown on or in the document. Enter last name, first name, middle initial. If military, show rank. The name of the principal author is an absolute minimum requirement.

DOCUMENT DATE Enter the date (month, year) of Establishment approval for publication of the document.

TOTAL NUMBER OF PAGES The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

NUMBER OF REFERENCES Enter the total number of references cited in the document.

PROJECT OR GRANT NUMBER If appropriate, enter the applicable research and development project or grant number under which the document was written.

CONTRACT NUMBER If appropriate, enter the applicable number under which the document was written.

ORIGINATOR'S DOCUMENT NUMBER(S) Enter the official document number by which the document will be identified and controlled by the originating activity. This number must be unique to this document.

9b. OTHER DOCUMENT NUMBER(S) If the document has been assigned any other document numbers (either by the originator or by the sponsor), also enter this number(s).

10. DISTRIBUTION STATEMENT Enter any limitations on further dissemination of the document, other than those imposed by security classification, using standard statements such as

(1) "Qualified requesters may obtain copies of this document from their defence documentation center."

(2) "Announcement and dissemination of this document is not authorized without prior approval from originating activity."

11. SUPPLEMENTARY NOTES Use for additional explanatory notes.

12. SPONSORING ACTIVITY Enter the name of the departmental project office or laboratory sponsoring the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document, even though it may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall end with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (TS), (S), (C), (R), or (U).

The length of the abstract should be limited to 20 single-spaced standard typewritten lines, 7½ inches long.

14. KEY WORDS. Key words are technically meaningful terms or short phrases that characterize a document and could be helpful in cataloging the document. Key words should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context.

END

FILMED

4-85

DTIC